D{5}1A{N8WS26{M7`62$WXC copy

JIANGXI AGRICULTURAL UNIVERSITY

**本 科 毕 业 论 文（设 计）**



**题目： 基于Redis缓存的生活服务平台管理系统的设计与实现**

**学 院： 软件学院**

**姓 名： 罗紫东**

**学 号： 20152028**

**专 业： 软件工程**

**班 级： 软件1501**

**指导教师： 袁露 职 称：**

**二0一九 年 二 月**

目 录

[摘要 I](#_Toc418435291)

[Abstract II](#_Toc418435292)

[1 绪论 1](#_Toc418435293)

[1.1 背景和意义 1](#_Toc418435294)

[1.2 现状和发展趋势 1](#_Toc418435295)

[1.3 研究方案 2](#_Toc418435296)

[2 技术简介 4](#_Toc418435297)

[2.1 Maven简介 4](#_Toc418435298)

[2.2 Resin4.0简介 4](#_Toc418435299)

[2.3 RabbitMQ简介 5](#_Toc418435300)

[2.4 Memcached简介 5](#_Toc418435301)

[2.5 本章小结 6](#_Toc418435302)

[3 需求分析 7](#_Toc418435303)

[3.1 技术可行性分析 7](#_Toc418435304)

[3.2 运行可行性分析 7](#_Toc418435305)

[3.3 视频网站媒资系统需求分析 8](#_Toc418435306)

[3.3.1 创建媒资信息 8](#_Toc418435307)

[3.3.2 视频媒资信息查询 8](#_Toc418435308)

[3.4 视频网站系统总体结构 9](#_Toc418435309)

[3.5 视频网站媒资系统用例图 9](#_Toc418435310)

[3.6 性能需求 10](#_Toc418435311)

[4 概要设计 12](#_Toc418435312)

[4.1 概要设计简介 12](#_Toc418435313)

[4.2 功能设计目标 12](#_Toc418435314)

[4.3 内部结构设计 12](#_Toc418435315)

[4.4 E-R图设计 13](#_Toc418435316)

[4.4.1 视频信息E-R图 14](#_Toc418435317)

[4.4.2 视频专辑信息E-R图 15](#_Toc418435318)

[4.4.3 视频媒资信息E-R图 16](#_Toc418435319)

[4.4.4 频道信息E-R图 16](#_Toc418435320)

[4.5 数据库表设计 17](#_Toc418435321)

[5 详细设计 20](#_Toc418435322)

[5.1 详细设计简介 20](#_Toc418435323)

[5.2 大型网站技术架构的演化设计 20](#_Toc418435324)

[5.2.1 视频网站媒资系统1.0版本 20](#_Toc418435325)

[5.2.2 视频网站媒资系统2.0版本 22](#_Toc418435326)

[5.2.3 视频网站媒资系统3.0版本 23](#_Toc418435327)

[5.3 视频网站媒资系统技术架构对比 27](#_Toc418435328)

[5.3.1 技术选择对比 27](#_Toc418435329)

[5.3.2 架构优缺点对比 27](#_Toc418435330)

[5.4 大型网站技术架构演化的设计理念 28](#_Toc418435331)

[5.4.1 价值观 28](#_Toc418435332)

[6 测试 28](#_Toc418435333)

[6.1 网站性能测试 28](#_Toc418435334)

[6.1.1 用户视角的网站性能 29](#_Toc418435335)

[6.1.2 开发人员视角的网站性能 29](#_Toc418435336)

[6.1.3 运维人员视角的网站性能 29](#_Toc418435337)

[6.2 测试环境 29](#_Toc418435338)

[6.3 性能测试指标 29](#_Toc418435339)

[6.3.1 基于top命令的硬件监控测试 30](#_Toc418435340)

[6.3.2 siege性能指标测试 31](#_Toc418435341)

[6.4 视频网站媒资系统接口性能测试对比 31](#_Toc418435342)

[6.5 视频网站媒资系统接口压力性能测试对比结果分析 34](#_Toc418435343)

[7 结论 35](#_Toc418435344)

[参考文献 36](#_Toc418435345)

[致 谢 37](#_Toc418435346)

# 摘要

近几年来，随着移动互联网的超快速发展，本地同城生活服务平台、同城生活服务app也越来越多，通过这些同城App软件，不仅让和用户们感受到了便捷、全面的生活服务平台，也给商家们提供了一个多元化的推广渠道，体验到了更接地气的商品售卖模式，可谓是一举两得。经过10年发展，中国人在网购方面，有了世界话语权。

开设网店，通常有两种途径，一种是在淘宝、阿里巴巴、百度有啊等网站申请使用，其优点是简单易用，搭建快速，缺点是自定义功能不强，在那里开店，只有选择，就好比大超市;另一种是选择一个合适的网上商店系统，自己搭建平台，就好比专卖店，没有价格竞争,深受广大用户青睐。

进入2019年这一行业也有了新的重要发展趋势。第一个是电商平台的影响会继续扩大。新的一年，电商平台将继续分解零售市场。作为零售商和品牌商的另一个销售渠道，电商平台变得越来有重要。但并不是每个零售商或品牌都会在平台上成功，对于那些想要把控客户体验和定价的零售商来说，平台销售可能无法像自有渠道那样把控各个环节。平台销售容易掀起价格战，导致利润率下降，价格策略失去控制。另外，平台的假货问题也是消费者们十分关注的问题之一。为了取得成功，零售商必须摸索出大型平台与垂直平台之间的差异。

关键词：互联网；生活服务；网站性能；高并发；架构；

# Abstract

Now, with the advent of the information age, the rapid development of the Internet, Internet content transmission tremendous changes have taken place in the early Internet with static text as the main content of public information, and the current transmission over the Internet is a lot of dynamic information. Meanwhile, with the popularity of the network, people get into the network life. Network traffic is increasing year by year, the scope of business increases every year, in the Internet market, more and more frequent shopping berserk. Service-oriented, high concurrency, high flow, large volumes of data network the new features are a test of the current Internet.

This paper discussed in detail and proved in a particular business scenario for performance comparison of different technical architecture show website system, as well as the advantages and disadvantages functional hierarchy, in which business scenarios are simulated multiple users simultaneous access to a video source Interface, the subject of this interface provides basic media information source to obtain the video information, the use of technical means to record the time of each user corresponding process all requests, and repeated in a number of concurrent access, the final analysis of the data through (e.g. connection the success rate of Availability, long response time average response time, the actual maximum number of concurrent concurrency, etc.) to prove the crucial role of technological innovation system architecture for a large site development of the Internet play.

**Key words:** internet; high concurrency; website performance; framework;

# 1 绪论

## 背景与意义

如今，随着信息时代的到来，互联网的迅猛发展，内容信息在互联网中的传输发生了颠覆性的变化，早期的互联网传输以静态、公共信息的文本为主要内容，而目前的互联网传输则是大量的动态信息，也包含了很多流媒体信息。同时，随着网络的普及，人们进入了互联网生活，加之5G时代的到来，真正的万物互联即将爆发。在使用网络时所需流量逐年递增，业务范围也在逐年扩大，在当前的互联网市场下，越来越频繁的购物疯抢不断出现，高扩展、微服务化、高并发、高流量、大数据量的网络新特性正在考验着当前的互联网。

为了解决高并发访问，大型网站都面临着海量数据处理，高可靠运行等一系列的问题和挑战，大型互联网公司在业务发展以及技术实践中提出了很多解决方案，各个公司公开的理论和实践资料越来越多也越来越成熟，各种开源的组件、工具、框架越来越丰富，相对于的社区维护越来越活跃，相应的都有了更成熟的解决方案。在不同的领域，在领先的互联网公司架构的研究，无论是国外还是国内，无论是Google、Facebook、亚马逊还是国内的BAT，都有它们独特和本地化的地方，为了实现高性能网站高可用性、易扩展性、可扩展性、安全性和其他技术指标体系，大型网站架构模型也在不断演化与完善。

作为一名刚入门的互联网开发工程师，能亲身经历一个网站从小到大的架构演化，从功能结构单一到复杂业务的开发过程是非常难得的，而往往只有这样才能深刻体会了解到符合特定业务场景的技术架构对于网站性能、安全性、扩展性是如此的重要，更需要我们积极地深入研究与探索。

## 现状和发展趋势

电子商务17年来取得了长足的发展，截止2018年6月，参与网购的网民规模达到8.02亿。电子商务对实体经济产生了重要影响。但95％的传统市场仍然不在电子商务领域。 O2O（ONLINE TO OFFLINE）是一种基于电子商务发展趋势的电子商务模式。 O2O是将在线消费者带到真正的商店--在线支付离线服务，然后上网享受服务。通过折扣（团购），提供信息和提供服务（预订），将离线商店的信息推送给在线用户，从而将他们转换为离线消费者客户。团购是020商业模式发展的一个特殊阶段。团购的本质是体验式营销。同时也是集团购买市场的爆发，促进了初级生活服务电子商务的消费体验，开启了中国电子商务的O2O时代。运用价值网络理论模型构建O2O商业模式生态学。O2O价值网络商业模式的要素包括消费者，商家，O2O运营平台和第三方支付平台。 O2O商业模式直接带来三个价值：信息、预约、优惠。在这个模型中，消费者处于整个价值网络结构的中心。通过对O2O商业模式的研究，发现O2O商业模式的特点、趋势、瓶颈和未来前景，促进了当地生活服务电子商务市场的成熟和发展。对O2O业务模型进行分类，追踪当前的O2O应用案例。分析出了四种不同类型的应用案例，例如易于使用的车辆，数千种产品，右道车品以及他们的客户。针对当前电信运营商面临的转型挑战，分析了电信运营商在O2O业务模式中的优势和机遇。 O2O本地生活服务电子商务是解决线下生活服务业的一种新的电子商务理论模型。这种商业模式和离线实体业务是一种良性和互动的商业模式，可以补充和赢得。由SOLOMO代表的O2O商业模式和移动互联网的结合将为未来的企业和消费者带来难以想象的消费者体验变化。 O2O模式面临数万亿元的市场机遇。未来，这个市场的领导者将超越腾讯，百度和阿里巴巴，成为中国互联网的新一代领导者。

随着互联网的发展，Web服务端开发技术也经历了几次大的变迁。早期的Web服务器只简单地响应浏览器端的请求，返回静态的HTML。随着CGI（Common Gateway Interface, 通用网关接口）技术的出现，服务端可以根据请求产生动态页面内容。PHP，及随后ASP, JSP的出现改善了这一局面，与CGI在程序中输出HTML流正好相反，开发人员可以在HTML中嵌入程序代码，这种模式被称为服务器页面模式。直到现在，Java语言成为主流，各种大型网站的架构都是依赖于Java强大的成熟 的体系，和Apache, MySQL, Linux共同组成一个强大的Web开发平台，被称作LAMJ。既然CGI程序擅长处理请求信息，而服务器页面擅长构造响应页面，那么将两者结合起来就是当下常见的MVC（模型-视图-控制器）模式，使用MVC模式可以很好地分离模型与视图，使二者完全解耦，互相影响降到最低。

模型和视图分离为系统开发维护带来了诸多好处，为目前Web开发流畅的分层架构模式奠定了基础。分层模式可以更进一步分离关注面和降低系统的耦合性，通过分层，隔离上层对下层的直接依赖，上层设计无需过多考虑下层实现；各层之间较少耦合，只要保持接口规范不变，各层可以随意替换和复用。Web开发中通常将服务端划分为三层：表现层，业务逻辑层和数据源层。表现层完成视图展现和用户交互；业务逻辑层实现系统的核心逻辑；数据源层负责数据存储，交换和通信。这种层次划分是逻辑上的，物理部署上多个层会作为一个应用部署在一起。

随着用户量的增加，业务功能的扩展逐渐演变完善，各种成熟的组件，工具，框架越来越丰富，各种理论体系也在不断完善与成熟，各大公司公开的理论和实践资料也越来越多，在各个领域都有比较成熟的解决方案，在这个过程中，开发模式、技术架构、设计思想也发生了很大的变化，就连技术人员也从几个人发展到一个部门甚至一条产品线。随之而来，高并发，高流量，数据量大的网络新特性正在考验着当前的互联网。因此，一个成熟的大型网站技术架构往往需要经过亿万用户的实时检测，互联网技术的发展也在朝着这个方向而不断迈进。

## 研究方案

由于硬件资源与人数有限，研究方式主要以理论+实验的方式，先详细阐述大型网站系统架构的演化对于开发模式、技术架构、设计思想的变化过程，重点针对于大型网站常见的分层，消息队列以及缓存的使用与选择，再以项目实战——视频网站媒资系统推导理论，根据大型网站系统架构的演化过程，设计出三种演化过程中的基础技术架构：

（1）架构一．JSP+JAVABean模式，使用最基础的jdbc提供服务。

（2）架构二．SSI垂直架构，使用常见持久化框架提供服务。

（3）架构三．分布式架构，将业务模块化，模块通过AOP或异步消息进行通信，使用基于memcached协议的缓存中间件提供服务。

根据三组技术架构开发的网站，对外提供一组接口api，使用压力测试工具Siege对该组接口进行压力测试，通过配置对一个WEB站点进行多用户的并发模拟访问，记录每个用户所有请求过程的相应时间，并在一定数量的并发访问下重复进行，最终通过分析各项数据（连接成功率Availability，平均响应时长Response time，实际最高并发数Concurrency等）来证明大型网站系统架构的技术革新对于互联网发展起到的至关重要的作用。

# 2 技术简介

## 2.1 Spring简介

Spring框架是一种轻量级的解决方案，为构建企业级应用提供一站式服务。Spring包括很多模块，但是并不意味着你要使用Spring就要引入它的很多功能，你可以根据自己的需求选择性的引入需要用到的模块（比如，你可以只使用Spring的IOC容器，而不用引入web模块）Spring是非侵入性的，这意味着业务逻辑代码可以不依赖于框架本身，开发人员可以专注于业务功能的实现，而不必关心框架内部的处理逻辑。

控制反转是Spring的核心思想，到底什么是控制反转，对什么的控制进行了反转？2004年，Martin Fowler在他的网站上提出了一个关于控制反转(IoC)的问题。“在控制的哪些方面发生了逆转?”。Fowler建议重新命名该原则，使其更易于解释，并提出了依赖注入。

控制反转是指，获取对象的方式进行了反转，传统的对象获取需我们在使用的时候主动去new，而现在不需要自己去new对象，需要用到对象时由spirng ioc容器帮你创建，创建对象的控制权由你自己变成spring ioc容器，这就是对象的控制权进行了反转。

依赖注入跟控制反转其实是从不同的方向表达同一种思想，依赖注入说的是对象的获取需要依赖spring ioc容器，当需要处理对象间的依赖关系时，spirng ioc容器会将需要的对象依赖关系注入到相应的对象中，不需要我们主动去创建对象。

## 2.2 Maven简介

Maven是一个跨平台的项目管理工具。作为Apache组织的一个颇为成功的开源项目，其主要服务于基于Java平台的项目创建，依赖管理和项目信息管理。maven是Apache的顶级项目，解释为“专家，内行”，它是一个项目管理的工具，maven自身是纯java开发的（ Theresult is a tool that can now be used for building and managing any Java-basedproject），可以使用maven对java项目进行构建、依赖管理。

使用maven带来的好处：（1）依赖管理，对jar的统一管理，节省空间。提到依赖，那什么是依赖呢？依赖是指：一个java项目可能需要使用到第三方的jar包才能运行，那么我们说这个java项目依赖于第三方jar包。比如：SSM(spring,spring mvc mybatis)，S2SH(Struts2 Spring hibernate）。（2）一键构建，如果是web项目构建的是war包，Java项目构建的jar包，具有编码->编译->测试->打包，运行打包部署。（3）跨平台。（4）应用在大型的项目中可以提高开发效率。商城项目分析：用户模块，订单模块，地址模块，支付模块；maven分模块开发，父子工程，servlet - service(业务)-dao-entity；互联网中项目：安装业务来分模块；传统项目：在JavaWeb项目中按层来分：entity，dao，service，web。

## 2.3 Redis简介

Redis是当前比较热门的NOSQL系统之一，它是一个开源的使用ANSI c语言编写的key-value存储系统（区别于MySQL的二维表格的形式存储。）。和Memcache类似，但很大程度补偿了Memcache的不足。和Memcache一样，Redis数据都是缓存在计算机内存中，不同的是，Memcache只能将数据缓存到内存中，无法自动定期写入硬盘，这就表示，一断电或重启，内存清空，数据丢失。所以Memcache的应用场景适用于缓存无需持久化的数据。而Redis不同的是它会周期性的把更新的数据写入磁盘或者把修改操作写入追加的记录文件，实现数据的持久化

Redis 优势

* 性能极高 – Redis能读的速度是110000次/s,写的速度是81000次/s 。
* 丰富的数据类型 – Redis支持二进制案例的 Strings, Lists, Hashes, Sets 及 Ordered Sets 数据类型操作。
* 原子 – Redis的所有操作都是原子性的，意思就是要么成功执行要么失败完全不执行。单个操作是原子性的。多个操作也支持事务，即原子性，通过MULTI和EXEC指令包起来。
* 丰富的特性 – Redis还支持 publish/subscribe, 通知, key 过期等等特性。

Redis与其他key-value存储有什么不同？

* Redis有着更为复杂的数据结构并且提供对他们的原子性操作，这是一个不同于其他数据库的进化路径。Redis的数据类型都是基于基本数据结构的同时对程序员透明，无需进行额外的抽象。
* Redis运行在内存中但是可以持久化到磁盘，所以在对不同数据集进行高速读写时需要权衡内存，因为数据量不能大于硬件内存。在内存数据库方面的另一个优点是，相比在磁盘上相同的复杂的数据结构，在内存中操作起来非常简单，这样Redis可以做很多内部复杂性很强的事情。同时，在磁盘格式方面他们是紧凑的以追加的方式产生的，因为他们并不需要进行随机访问。

## 2.4 MySQL简介

可以公平地说，MySQL是最流行的开源数据库。它有一个非常庞大的安装基础和数量的用户。让我们看看MySQL之所以如此流行的原因，以及它目前的状况，也许还会触及到它的一些未来(尽管预测未来是非常罕见的)。查看MySQL的客户领域，包括facebook、flickr、Adobe(InCreativeSuite 3)、Drupal、Digg、LinkedIn、Wikipedia、ebay、youtube、google AdSense，很明显，MySQL是无处不在的。当你登录到你的热门论坛(由Bulleting驱动)或博客(由WordPress驱动)时，很可能它有MySQL作为它的后端数据库。传统上，MySQL的两个特点，简单的使用和性能，使它得到了这样的普及。除此之外，在非常广泛的平台(包括windows)和内置复制(为只读客户端提供了一个简单的扩展解决方案)上的可用性，给用户带来了更多的吸引力和产品部署。MySQL的简单性有很简单的证据：在15分钟或更短的时间内，您就可以安装、拥有一个可工作的数据库，并开始运行查询和存储数据。从早期阶段起，MySQL与大多数流行的Web开发语言都有一个很好的接口。PHP和Perl，以及Java和ODBC连接器。MySQL中有两个最著名的存储引擎：MyISAM和InnoDB(我这里不讨论NDB集群；这是一个完全不同的故事)。MyISAM是默认的存储工程，历史上它是最古老的，但InnoDB兼容ACID，并提供事务、行级锁定、MVCC、自动恢复和数据损坏检测。这使得它成为您希望为应用程序选择的存储引擎。另外，还有第三方事务存储引擎pbxt，它的特性类似于InnoDB，它包含在MariaDB发行版中。

InnoDB存储引擎支持事务，其设计目标主要面向在线事务处理(OLTP)的应用。其特点是行锁设计、支持外键，并支持类似于Oracle的非锁定读，即默认读取操作不会产生锁。从MySQL数据库5.5.8 版本开始，InnoDB 存储引擎是默认的存储引擎。

InnoDB存储引擎将数据放在一个逻辑的表空间中，这个表空间就像黑盒- -样由InnoDB存储引擎自身进行管理。从MySQL4.1 (包括4.1) 版本开始，它可以将每个InnoDB存储引擎的表单独存放到- -个独立的ibd文件中。此外，InnoDB 存储引擎支持用裸设备(row disk)用来建立其表空间。

InnoDB通过使用多版本并发控制(MVCC)来获得高并发性，并且实现了SQL标准的4种隔离级别，默认为REPEATABLE级别。同时，使用一种被称为next-keylocking的策略来避免幻读(phantom) 现象的产生。除此之外，InnoDB 储存引擎还提供了插入缓冲(insert buffer)、二次写(double write)、自适应哈希索引(adaptive hashindex)、预读(read ahead)等高性能和高可用的功能。

对于表中数据的存储，InnoDB 存储引擎采用了聚集(clustered) 的方式，因此每张表的存储都是按主键的顺序进行存放。如果没有显式地在表定义时指定主键, InnoDB 存储引擎会为每一行生成一个6字节的ROWID,并以此作为主键。

## 2.5 AngularJS简介

AngularJS是一个开发动态Web应用的框架。它让你可以使用HTML作为模板语言并且可以通过扩展的HTML语法来使应用组件更加清晰和简洁。它的创新之处在于，通过数据绑定和依赖注入减少了大量代码，而这些都在浏览器端通过JavaScript实现，能够和任何服务器端技术完美结合。

Angular是为了扩展HTML在构建应用时本应具备的能力而设计的。对于静态文档，HTML是一门很好的声明式的语言，但对于构建动态WEB应用，它无能为力。所以，构建动态WEB应用往往需要一些技巧才能让浏览器配合我们的工作。

通常，我们通过以下手段来解决动态应用和静态文档之间不匹配的问题：

* 类库 - 一些在开发WEB应用时非常有用的函数的集合。你的代码起主导作用，并且决定何时调用类库的方法。例如：jQuery等。
* 框架 - 一种WEB应用的特殊实现，你的代码只需要填充一些具体信息。框架起主导作用，并且决定何时调用你的代码。例如：knockout, ember等。

Angular另辟蹊径，它尝试去扩展HTML的结构来弥合以文档为中心的HTML与实际Web应用所需要的HTML之间的鸿沟。Angular通过指令（directive）扩展HTML的语法。例如：

* 通过{{}}进行数据绑定。
* 使用DOM控制结构来进行迭代或隐藏DOM片段。
* 支持表单和表单验证。
* 将逻辑代码关联到DOM元素上。
* 将一组HTML做成可重用的组件。

## 2.5 本章小结

本次研究主要为生活服务平台的后台管理系统，和传统的后台管理系统类似，都是以数据统计为主，提供灵活的可操作性。在架构技术的选择上，后台架构已经有很多完善稳定的体系，为提高可视化和操作的效率，选取了Spring主流MVC框架和主流的Redis高速缓存服务器与经典的MySQL数据库作为重点研究对象。采取前后端分离技术搭配AngularJS。Linux下搭建服务容器Tomcat，并部署项目，线上环境测试、日志监测等。

# 3 需求分析

互联网生活服务平台和普通的线上服务不一样，介于纯电商贸易和线下本地零售业之间的一种互联网经济模式，适用于本地化的超市、水果商店、小卖铺等等小型贸易体。打通快速的同城本地交易，生活服务平台移动端应用即可满足这种市场，平台以会员管理系统、商家管理系统、渠道管理系统为基础，以手机客户端为重点、门户网站为辅助。生活服务平台的后台管理系统也就应运而生。

和现存的本地外卖服务类似，但是所涵盖的不仅仅是食物外卖的单一商品，包含超市大量商品种类。后台管理系统将移动应用所产生的信息统计到数据库中，并提供灵活的需求视图，为满足大量的市场运营人员分析数据，装配高速缓存技术，准确无误的提供数据保障。

## 3.1 技术可行性分析

可行性分析所要描述的任务,并不是所研究问题都需要有简单明显的解决办法，事实上，许多问题不可能在预定的系统规模之内解决。如果问题不是一个可行的解决方案，然后开发项目的任何时间，资源，成本，人力和资金的不必要的浪费。可行性研究的目的是在尽可能短的时间内确定问题是否可以用最低的成本解决。因此，可行性研究的目的不是解决问题，而是确定问题是否解决。

作为一个服务于交易平台的后台管理系统，从技术可行性分析，随着互联网技术的不断拓展，业务的逐渐扩张，越来越多的传统型行业也在逐步向互联网转型，后台管理系统的服务压力会变得越来越大并且服务量也会逐渐增加，服务的解耦和分离以及降低持久层的压力尤为重要，所以采用高可用高并发量的技术架构的实现是可行的。

## 3.2 运行可行性分析

部署生活服务平台后台管理系统，最重要的是最关键的是硬件资源，每日的活跃用户都会产生大量的数据信息，充足的硬件设备尤为重要，虽然说对于目前架构而言，可以进行横向无线扩容，但是资源的紧凑实用也是有必要的，相应的硬件资源如下：

运行环境：

1. 操作系统：Ubuntu 16.4
2. 数据库：MySQL5.7
3. 开发工具：IDEA
4. 缓存服务器：Redis
5. 运行服务器：Tomcat

硬件环境：

1. 处理器：Intel i5
2. 内存：16G
3. 硬盘容量：1T
4. 其它：鼠标、键盘、显示器、服务器主机

## 3.3 生活服务平台后台管理系统需求分析

### 3.3.1 权限管理

1. 功能需求：菜单管理

描述：对后台界面对特定用户显示的菜单管理，并可以对菜单进行增加、修改、删除等操作

1. 功能需求：角色管理

描述：对后台用户的角色进行管理，角色所授予的菜单进行管理

### 3.3.2 会员管理

1. 功能需求：会员管理

描述：对生活服务平台的会员用户进行管理，对用户进行增加、修改个人信息、注销用户

### 3.3.2 门店管理

1. 功能需求：门店管理

描述：入驻生活管理平台的商家门店进行统一管理，并可查找该门店下所有商品信息，以及门店的经纬度信息，可以在百度地图上精确显示

### 3.3.3 订单管理

1. 功能需求：订单管理

描述：生活服务平台用户在平台上进行所有交易所产生的订单的显示，跟踪订单的状态信息，并提供信息更新、错误信息更正的功能，运营人员、客服可以对客户反馈的订单号给用户提供订单详细信息反馈。

### 3.3.4 板块管理

1. 功能需求：板块管理

描述：对生活管理平台上架的板块进行后台管理，可以人工上线活动板块，人工下架过期活动等

### 3.3.5 数据统计

1. 功能需求：数据统计

描述：通过大数据方对后台数据统计后写入后台系统的数据库，然后通过后台管理平台可以进行相应的图形化展示，可以清晰客观的分析用户的交易画像。

### 3.3.6 系统设置

1. 功能需求：系统设置

描述：对后台管理系统进行相关的系统设置

## 3.4 基于Redis缓存的生活服务平台管理系统总体结构

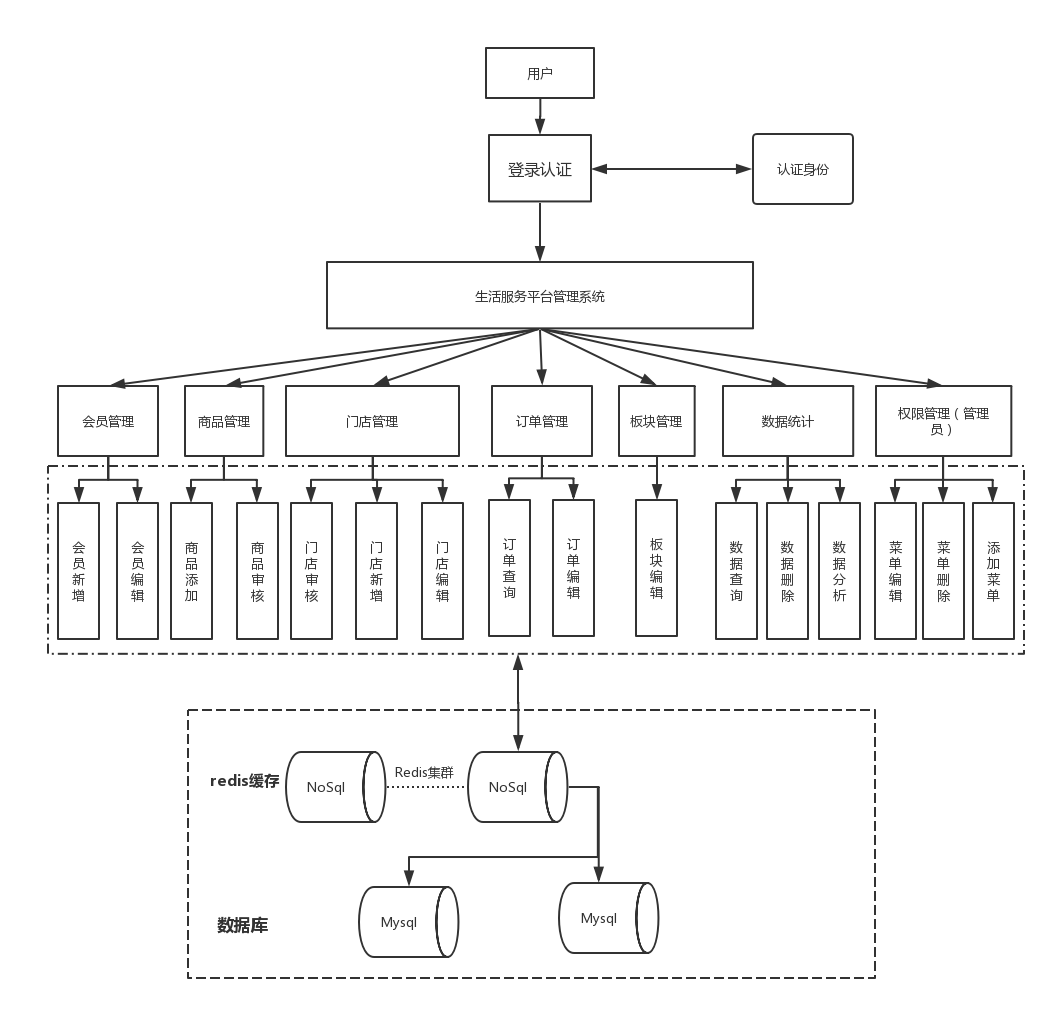


图3-1 基于Redis缓存的生活服务平台管理系统总体结构图（待修改）

需要体现总体，不能有过多内部结构

## 3.5 基于Redis缓存的生活服务平台管理系统用例图

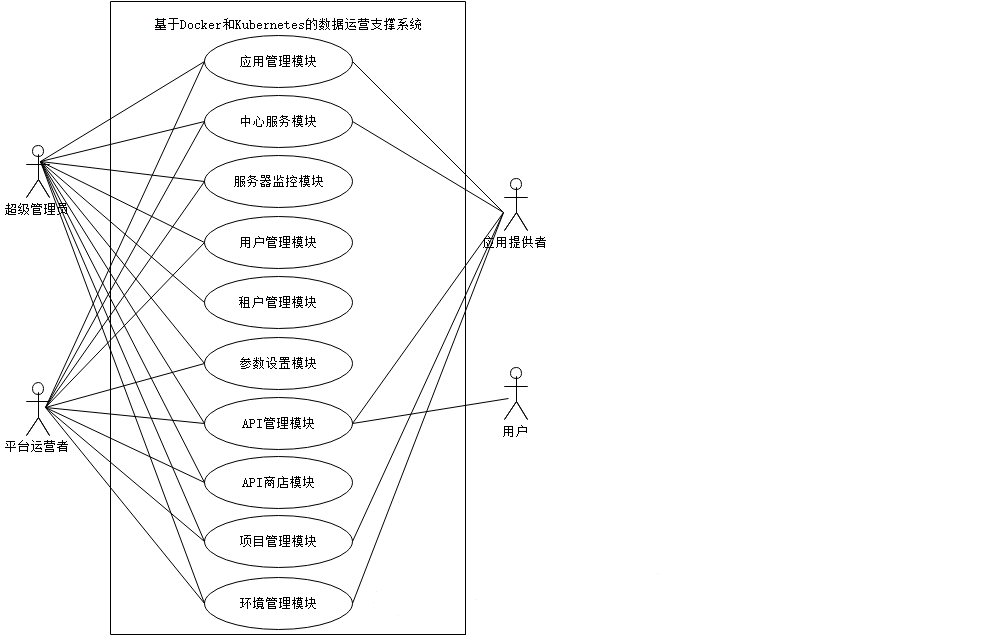


图3-2 基于Redis缓存的生活服务平台管理系统用例图

TODO删除，换成自己的

# 4 概要设计

## 4.1 概要设计简介

概要设计是联系需求分析与详细设计两阶段之间不可或缺的一个重要阶段，概要设计的主要任务是把需求分析得到的系统用例图转换为软件结构和数据结构。软件结构则主要以总体架构图来展示，表现出系统所要完成的绝大部分功能，而数据结构则以E-R图以及数据库设计为主，明确系统数据存储与设计。

## 4.2 功能设计目标

本次课题主要为运用三种不同技术 架构的系统性能对比测试，但其业务均为视频网站媒资系统，运用的数据结构与数据源相同，因此，概要设计阶段统一阐述，不再分述，主要表现出媒资系统绝大部分功能以及E-R图等。

## 4.3 内部结构设计

生活服务平台后台管理系统内部结构如图xxx所示：

## 4.4 E-R图设计

E-R图也称实体-联系图(Entity Relationship Diagram)，用来描述现实世界的概念模型。其表示方法为：

（1）实体型：矩形表示，矩形框内为实体名，代表所要描述的某一实体；

（2）属性：椭圆形表示，框内写上具体属性名称

（3）联系：菱形表示，菱形框内为联系的名称，代表实体与实体之间的联系，同时在联系的相应边旁标上联系的类型，有1:1 1:n n:m。

生活服务平台后台管理系统E-R图如图xxx所示：

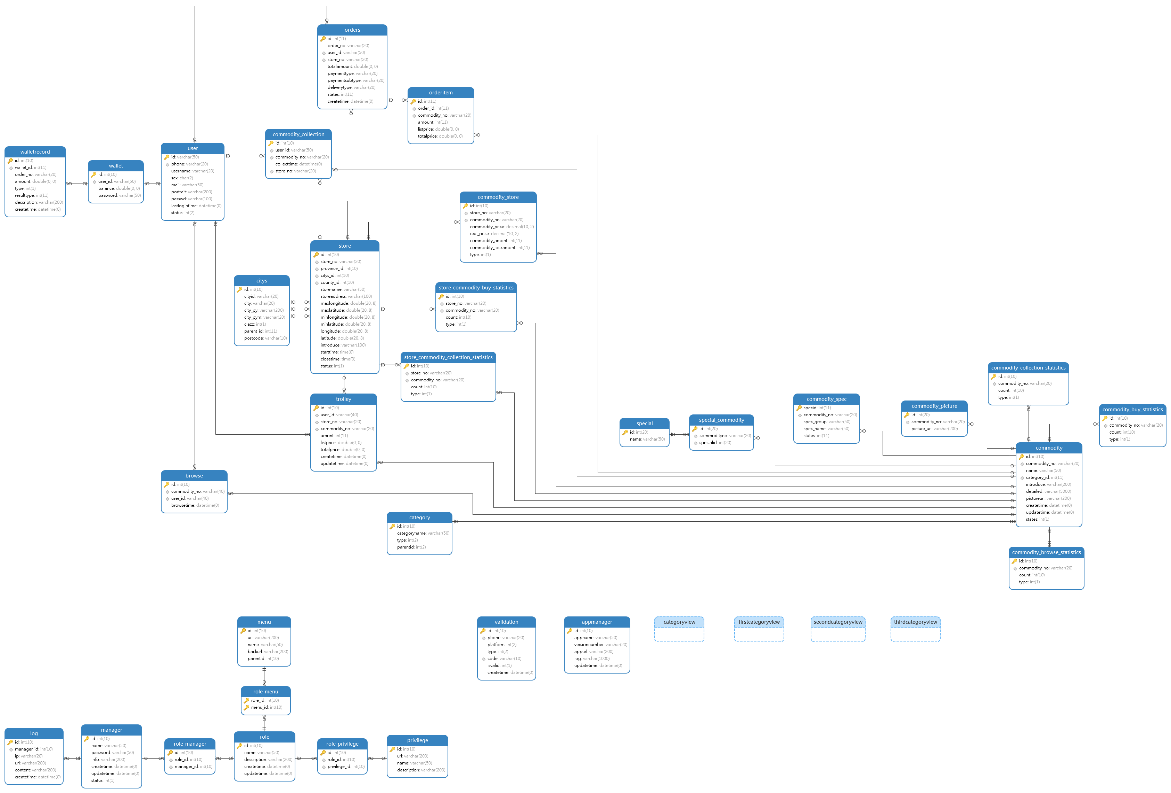
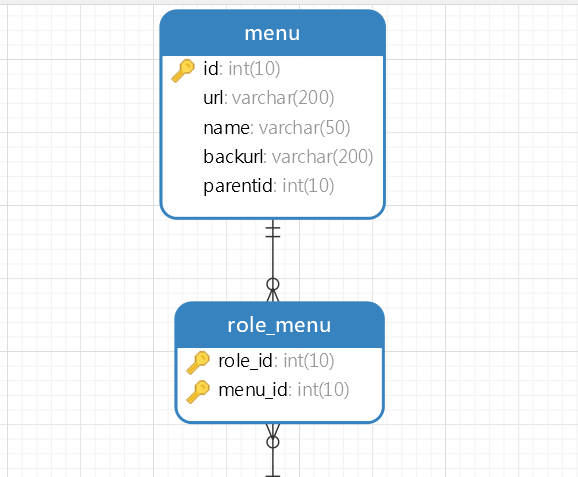
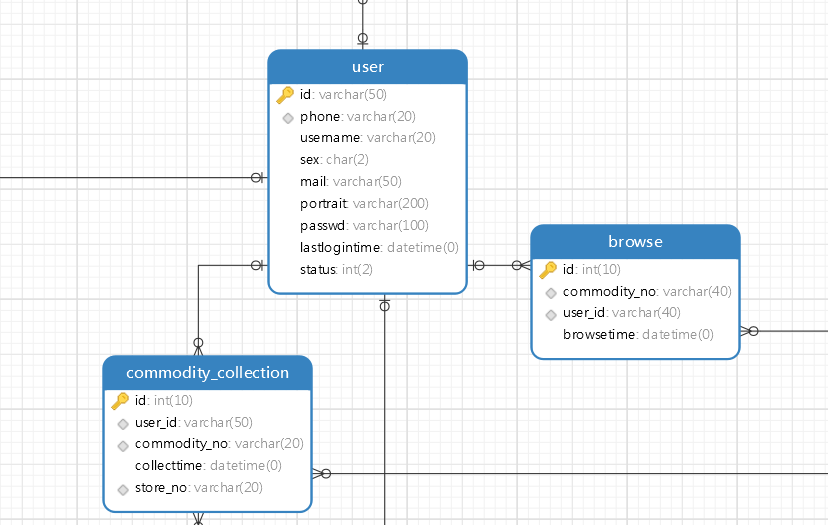


图4-2 基于Redis缓存的生活服务平台管理系统E-R图

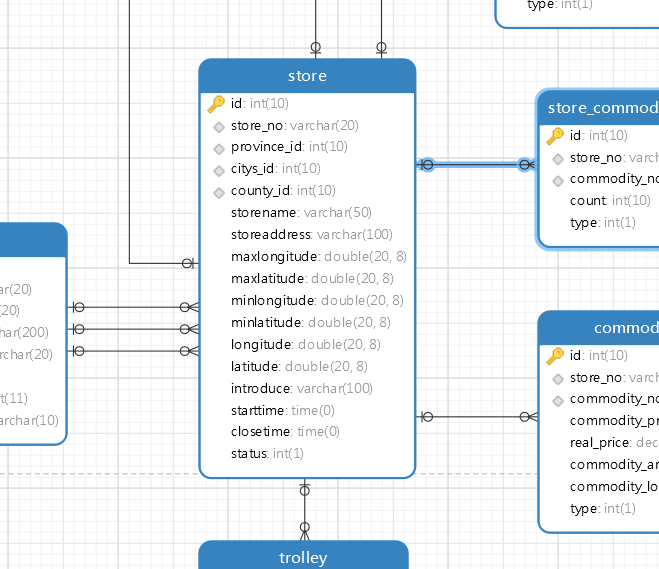
### 4.4.1 菜单权限E-R图



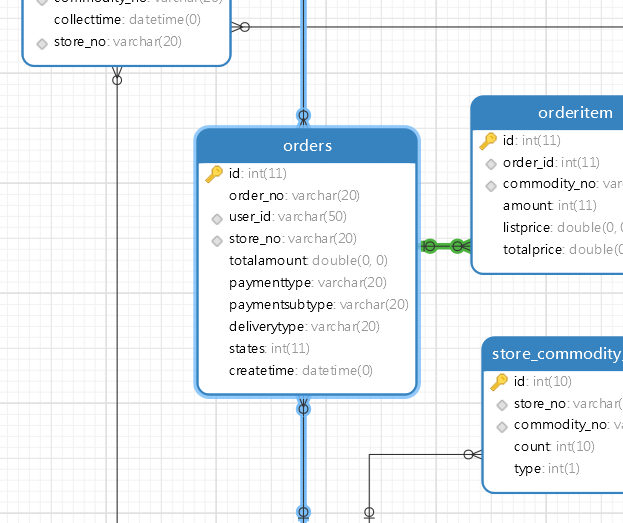
### 4.4.2 会员信息E-R图



### 4.4.3 门店信息E-R图



### 4.4.5 订单信息E-R图

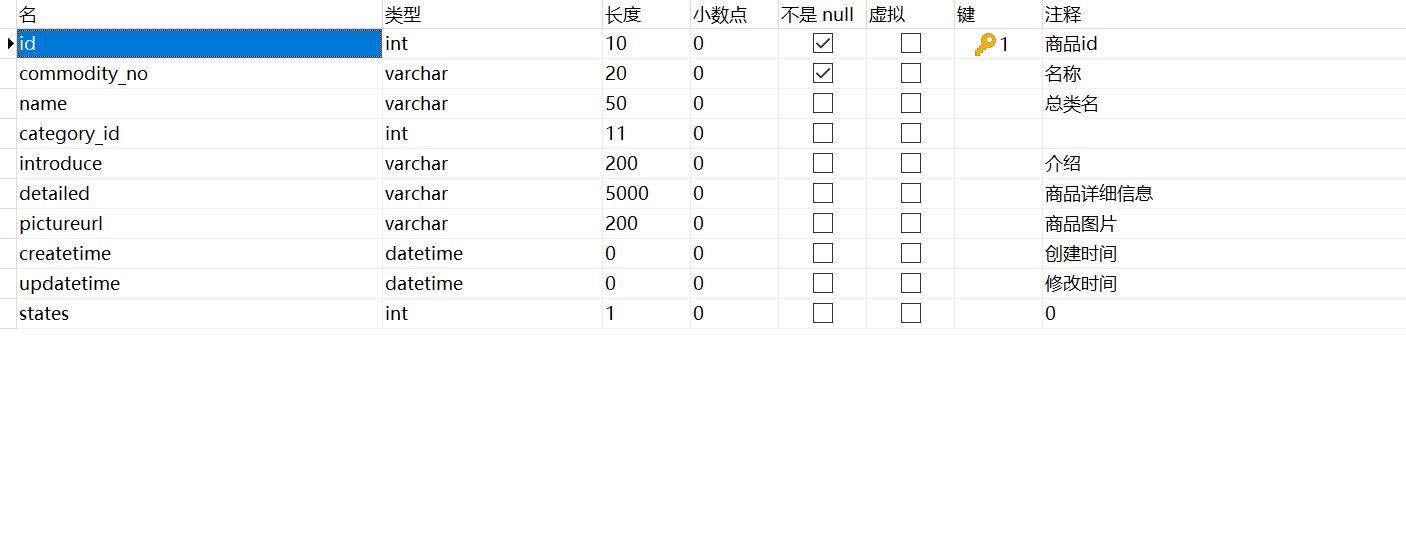


## 4.5 数据库表设计

门店信息表，如表所示：



商品表：



订单表：



# 5 详细设计

# 6 系统实现

# 7 程序测试与评价

## 7.1 程序调试

在设计系统的过程中，存在一些错误是必然的。对于语句的语法错误，在程序运行时自动提示，并请求立即纠正，因此，这类错误比较容易发现和纠正。但另一类错误是在程序执行时由于不正确的操作或对某些数据的计算公式的逻辑错误导致的错误结果。这类错误隐蔽性强，有时会出现，有时又不出现，因此，对这一类动态发生的错误的排查是耗时费力的。

## 7.2 程序的测试

### 7.2.1 测试的重要性及目的

1. **测试的重要性**

在软件生命周期中，软件的测试占据着一个重要的位置，在传统的瀑布模式中，软件测试学只是处于在运行维护阶段之前，是软件产品交付给大众用户使用之前用以保证软件质量重要的一个方法。近来，软件工程界中更加趋向于一种新观点，即认为软件每一阶段中的生命周期都理应包含测试，因此使检查本阶段的成果能不能接近预期的结果目标，进而尽可能早一点发现到错误并加以改正，如果测试不是在早期阶段进行的，那么错误将延时扩散，最后常常会导致成品测试面临巨大的困难。

对于软件来讲，事实上，无论采用任何方法或者任何技术，系统中仍然会出现错误。采用新的开发方式、先进的语言、完善的开发过程，可以最大限度的降低错误的引进，但还是不能完全的杜绝软件中有可能出现的错误，这些引入的错误需要通过测试找出来，系统中错误的密度也需要通过测试来进行估量。测试是所有学科的工程的最基本的组成单元，是软件开发的一个很重要的部分。自有设计程序的那天起测试就一直伴随着。统计表明，在典型的软件开发项目中，软件测试工作量往往占软件开发总工作量的40％以上。而在软件开发的总成本中，用在测试上的开销要占30％到50％。如果把维护阶段也考虑在内，讨论整个软件生存期时，测试的成本比例也许会有所降低，但实际上维护程序的工作相当于二次开发，甚至是多次开发，其中一定又包含了很多测试的工作。

在实践中，测试软件非常困难，从而常常使我们敷衍了事或望而却步，这是因为对测试还是存在了一些错误的看法和不正确的态度，这包括:

1. 难以给测试的工作人员某种成就感; 认为测试工作并不像编码和设计那样容易进行发展
2. 在测试的过程中发现错位其实是一种对责任者工作的否定; 测试是以发现软件的错误为目标的，这是非建设性的，甚至还是具有破坏性的，
3. 测试的工作过程非常枯燥无味，因此不足以引起我们的兴趣和爱好;
4. 测试工作需要是细致而艰苦的进行;
5. 一般来说，我们都对自己编写出来的程序比较盲目的自信，在发现到错误的时候，可能会顾虑别人对我们自己开发的能力的看法。

以上这几个观点对测试系统的工作进行是尤为不利的，我们必须要端正态度、澄清认识，才可能将系统产品的质量加以提高。

1. **测试的目的**

如果测试程序的目的是为了尽可能全面地找出系统中的错误，那么测试就必须直观的针对软件较为复杂的部分或者是前期出现错误比较多的位置。

1. 系统的测试是为了检查出错误而执行程序的运行过程;
2. 测试则是为了证明程序中有错误，而不是用来证明程序没有错误;
3. 每一个好的测试的用例是因为它能检查到至今未出现的错误;
4. 一个很成功的测试就是检查出了至今未能发现的错误的测试。

这种观点可以提醒人们测试要以查找错误为中心，而不是为了演示软件的正确功能。但是仅凭字面意思理解这一观点可能会产生误导，认为发现错误是软件测试的唯一目，查找不出错误的测试就是没有价值的，事实并非如此。

首先，测试并不仅仅是为了要找出错误。通过分析错误产生的原因和错误的分布特征，可以帮助项目管理者发现当前所采用的软件过程的缺陷，以便改进。同时，这种分析也能帮助我们设计出有针对性地检测方法，改善测试的有效性。其次，没有检查出错误的测试也有它的价值，完整的测试是测试质量评定的一种方式。

### 7.2.2 测试的步骤

### 7.2.3 测试的主要内容

### 7.2.4 测试用例

## 7.3 系统的特点和优点

## 7.4 系统存在的不足和改进方案

## 7.5 设计收获和心得

# 8 结论

~~大型网站技术架构的核心价值并不是从无到有搭建一个大型网站，而是能够伴随小型网站业务的逐步发展，慢慢地演化成一个大型网站。一方面随着互联网的高速发展，越来越多新的软件技术和产品从互联网公司诞生，挑战传统软件巨头的江湖地位，另一方面却是中小网站十几年如一日地使用LAMP技术（Linux+Apache+MySQL+PHP）开发自己的网站，因为LAMP既便宜又简单，而且对付一个中小型网站绰绰有余。因此，一味追随大公司，大型网站的解决方案，为了技术而技术以及企图用技术解决所有问题都是网站架构设计的误区，我们都需要在网站架构中去避免，去解决。~~

~~另一方面，大型网站的架构演化方案已经非常成熟，各种技术方案也逐步产品化，许多小型网站已经慢慢不需要再经历大型网站经历过的架构演化之路就可以逐步发展壮大，因为现在越来越多的网站从建立之初就是搭建在大型网站提供的云计算服务基础之上，所需要的一切技术资源：计算、存储、网络都可以按需购买，线性伸缩，不需要自己一点一点地拼凑各种资源，综合使用各种技术方案去完善自己的网站架构了。所以，这也是本人选题的真正原因，能亲身经历一个网站从小到大的架构演化对于未来软件开发之路肯定是大有裨益的。~~

# 参考文献

[1]许晓斌，Maven实战[M].北京：机械工业出版社，2011.

[2]李智慧． 大型网站技术架构：核心原理与案例分析[J]．电子工业出版社，2013-9-1，218

[3]李平．大型网站系统架构的演化． http://www.cnblogs.com/leefreeman/p/3993449.html 2014-09-26．

[4]陶辉，深入理解Nginx[M].北京：机械工业出版社，2014.

[5]郭峰，Spring从入门到精通[M].北京：清华大学出版社，2006.

[6]Charles A Bell，Expert MySQL[M].北京：人民邮电出版社，2010.

[7] Bruce Eckel著，Thinking In Java[M].北京：机械工业出版社，2006.

[8]Michael Kofler，MySQL 5权威指南（第三版） [M].北京：人民邮电出版社，2006.

[9]蒋文旭，辛阳（北京邮电大学计算机学院信息安全中心，北京 100876）大型高并发web应用系统架构分析与设计 http://www.doc88.com/p-4179020971453.html

[10] 杨光辉．互联网型系统架构的演进． http://www.csdn.net/article/2013-08-27/2816716 2013-08-29．

# 致谢

~~经过一段时间的努力，本次论文终于圆满完成，平时在工作中与同事交流，无意中受到启发，才选定了本次的课题。本次课题，对于我这个刚入门互联网企业的初级程序员来说，是一个较大的挑战，而且主要以理论为主，并不是之前所做的偏应用型的增删改善项目，在定题之后，阅读并尝试学习了大量技术，初步对架构有了一定的认识，也明白了架构是随业务而发展的，并不能一味追求技术的狂热。~~

~~其次，我要感谢导师邓泓副教授的帮助，开题之初就给了我很多的宝贵意见，让我能顺利的完成此次论文的撰写，另外也要感谢同学，同事们的帮助和指导，是你们的耐心讲解与分析才让我更好的完成了论文。~~

~~最后，也要感谢自己，因为此次课题的研究历经磨难和考验，很多次都想过放弃而去随意完成一些简单的系统，但是为了能亲身经历一次大型网站技术架构的演化过程，并且自己搭建各种软件环境，为自己的坚持而感到骄傲，希望在未来的软件开发道路上能继续坚持下去，努力地成为一名优秀的互联网开发人员。~~