摘 要

随着移动互联网的迅猛发展，LBS 行业以史无前例的速度增长，各类 LBS 产品层出不穷：基于地理位置的社交，基于地理位置的游戏，基于地理位置的电子商务等。然而，目前大多数 LBS 应用对服务模式并不是很清晰。由于技术的不成熟和移动手机本身的特点限制了 LBS 应用的进一步发展，所以越来越多的应用开始寻求新的技术突破口。

Push 技术是一种基于客户服务器机制，由服务器主动的将信息发往客户端的技术，其本质在于让信息主动去寻找用户，因此其优势在于信息的主动性和及时性。采用 Push 技术来获取移动应用数据，无论对于用户还是内容提供商来说都能使数据内容得到很好的利用。

本文就是根据目前 LBS 的发展现状，将 LBS 现有技术与 Push 技术相融合，设计并实现了一个基于地理位置的即时信息推送服务的 LBS 系统（Shopylife），改变了用户的移动信息获取方式，增强了商家和顾客的即时互动。论文中首先分析了推送型 LBS 应用的实现方式。然后对系统的架构和总体流程及设计方案进行了描述。最后对系统的各个功能模块开发和实现进行了详细的描述。采用了 LAMP 技术对系统的核心功能进行了封装，并为 IOS 移动应用开发提供接口调用。

关键词：位置服务，移动应用，信息推送

ABSTRACT

With the rapid development of the mobile Internet, the LBS industry has an unprecedented growth, such as location based SNS, location based game, location based e-commerce.At present, however, most of LBS application's service mode is not very clear. As the immature technology and the characteristic of mobile phone limit the further development of the LBS application, so more and more LBS application begin to seek new technology breakthrough.

Push technology is a client server based mechanism, which means the server send the information to the client initiatively. The essence of push technology is to make information easy for customers to get, and its advantages incarnate the activity and personalization of information. Using push technology to retrieve application data, both for the user or the content providers can make the data content in good use.

This thesis, according to the current situation of the development of LBS, combined push technology with LBS technology to design and implement a location based instant information push service of system(shopylife),which changed the mobile users' way of obtaining information and enhanced the interaction of the merchants and customers. This thesis has first analyzed the implement ways of push LBS application. And then it described the system structure and the overall process and design proposal. In the end, it described each function module of system's development and implement in detail. It used LAMP technology to encapsulate the core function of the system, and provided interfaces for IOS mobile application development.

Keywords: Location Based Service, Mobile App, Information Push

目 录

第一章 绪论

1.1 引言

伴随着移动互联网用户的高速增长，移动互联网市场规模也在不断扩大，让 LBS 这块市场吸引了众多国内互联网巨头的关注，并纷纷涌现出一批如街旁网、开开网等创业公司，目前新浪推出了微领地、网易推出了八方，盛大有切客，百度、人人网等均推出了自己的 LBS 服务。早在 6、7 年前，就有一批类似 Foursquare 的足迹类网站出现，当时大多与旅游相结合，一直没有获得显著的效果。Foursquare 模式[1-3]的出现，将典型的 WEB2.0 应用网站混搭在一起，把传统互联网和移动互联网进行了很好的融合[4]-[6]。在近期的 LBS 大会上，各代表纷纷表示，LBS[18]-[23]需要从 Check-in 模式转到 O2O 模式[7]-[8]，从一种娱乐性的应用转移到实质性的应用上来。Shopylife 就是一个融合了传统互联网和移动互联网的 LBS 系统，把线上的优势和线下的资源进行了有效的整合。

1.2 项目背景

论文选题来自本人从事开发的一个项目：Shopylife(http://www.shopylife.com 购生活)。该项目由 iPhone 客户端，Web 系统两大部分组成，其中 Web 系统部分又分为 Web 前台系统，商家后台系统，以及 API 系统，我在整个项目里面主要从事 Web 系统的开发，并兼顾做了些 iPhone 的开发工作。移动手机客户端已经成功上线。Web 系统还处于邀请码注册状态。

Shopylife 是一个基于地理位置的以社会化购物分享和优惠券为即时信息的 Push 型系统，当用户注册成为网站的会员后，可以选择是否绑定其它的社会化账号（比如 Foursquare、Facebook、Twitter、Brightkite 等），然后我们会根据用户的地点选择设置推送信息的距离，由于初始网站的场所数据库不是很全，一般默认为 10 千米。通过下载网站的移动手机端应用，用户可以用手机拍下自己想要买的商品，并且可以为这些图片标上标签，然后分享给好友或者同步到其他连接网站上去，用户也可以通过此操作获取到更多的优惠。同时，用户也能及时地看到好友的分享或者网站推送的附近的热门商品。

整个项目历经了 3 个时期：

（1）Lywoo 时期，这是 shopylife 最原始的版本，中文名可叫做“礼物”，在 2010 年 10 月份的时候，当时看到 Foursquare 很火，网站最初的想法也是做成一个纯粹性签到的这样一个网站。

（2）Couponsensor 时期，中文名叫“优惠券感应器”，随着用户对单纯的签到逐渐失去了兴趣，我们立即改变了思路，引入一些激励机制，比如签到得勋章，签到达到一定的次数就将获得某些折扣。

（3）Shopylife 时期，中文名叫“购生活”，与之前的两个版本相比，目前这个版本已经算相当稳定的版本了，功能需求逐渐成熟，同时又有清晰的运作方式和盈利模式。目前 iphone 应用已经登上应用市场，网站还在紧张的内测之中。

1.3 Push 技术简介

Push 技术并不是最近才产生的技术，早在 1996 年 PointCast Network 公司就提出了用 Push 技术来获取网上信息，其目的是为了提高基于计算机网络的信息获取效率[15]。使用户不再是主动的去寻求信息，而是像收听广播看电视一样，被动地接受信息。但在后来的实际应用中，Push 技术却慢慢的淡化了，主要因为人们对自己定制的所有信息并不一定是非常喜欢和迫切需要的，人们对于信息的准确性远远高于信息的及时性要求，另一方面，在 PC 上不能确保用户随时随地都能收到推送的信息。

但是到了移动互联网时代，人们可以利用移动终端接收信息。在智能手机上面，想看股票有与股票相关的应用，看新闻有看新闻的应用，很多事情不用通过在浏览器中搜索来完成。另一方面，对于那些信息更新速度快，信息产生频度高而且用户需求相对较为稳定的移动数据内容而言，采用 Push 方式无论对于用户还是内容提供商来说都能使数据内容得到很好的利用，尤其是基于地理位置的实时推送（Push）技术将成为一种更加精准的营销方式。

Push 技术的本质在于让信息主动去寻找用户，因此其优势在于信息的主动性和及时性[16]。相对于 Pull 型服务而言，Push 技术具有以下特征[16][17]：

（1）主动性，Push 型服务器会在用户不发出任何信息请求的情况下，及时主动地告知用户，因此，用户可以主动获取到自己想要的信息。

（2）个性化，用户可以定制时间，推送内容、推送频率等参数，Push 型服务器会为不同的用户推送不同的信息内容。

（3）智能化，根据一些特定的算法，分析出哪些信息是用户觉得不需要的信息，然后过滤掉这些不重要的信息。

1.4 Push 型 LBS 应用分析

位置服务(Location based service，LBS)，又称定位服务，是指通过各种定位技术来确定移动用户的位置，从而为用户提供相关的服务信息[9]-[11]。LBS 应用主要分为两种类型：Pull 型和 Push 型[12][13]，Pull 型 LBS 应用是由用户向 LBS 内容提供商请求与位置相关的服务，而 Push 型 LBS 应用是由内容提供商主动对用户提供相关信息的 LBS 服务[14]，如与位置有关的公交信息服务，基于位置的出租车服务，基于位置的旅游服务，以及基于位置的优惠券服务等。

在 LBS 应用中，推送服务体现出了更大的优势。其实 Push 有两个含义，一个是定时通知，另外一个是实时消息。用户对定时通知的要求并不是及时的，当有新的内容产生，可以等待很久，在某个特定的时候通知用户阅读，比如视频、新闻等推送应用，用户并不希望一有新的新闻就马上推送到手机上，在特定的时间推送不会影响用户的阅读。而实时消息的推送一般对信息获取的要求非常及时，一旦服务器上有更新，就马上推送给手机用户，让用户阅读，这就使得基于推送的实时互动成为可能，例如邮件推送应用，用户希望一有新的邮件便立即通知，而不是过了很长一段时间才收到。由于移动终端的电池消耗快，移动网络流量费用高，移动设备的操作便捷性差。因此，移动智能手机就可以通过 Push 技术来获取数据。

在 Push 应用中，只要满足用户预先设定的条件，就将触发 Push 服务器发送内容给用户，服务器被触发以后，客户端就会收到从服务器发送过来的推送内容，而触发的条件也可以是多种多样的，可以是定制的位置距离，也可以是预先设定的时间等。

实际上，移动设备的 Push 机制[21]-[30]，基本上有 3 种做法：

（1）采用 Socket 常连接机制。采用这种方式，其实要求 TCP/IP 一直保持连通的状态，对于移动网络来说的话，这种方式耗电量太大。同时，移动终端也必须得是 Socket Server，而且还需要处理一些问题，比如 IP 注册、断线重连等。

（2）控制信道 Push。不是利用 TCP/IP 连接的方式，而是利用底层的移动通信控制信道进行推送，就是呈现我们手机是哪家信号，是否有电话呼入，是否注册在网的那个信道，就像短信的推送提醒机制一样，也是走的这个信道。由于这

是走的底层移动通信的控制信道，所以它不会耗电，但是这样的做法，是必须要和移动运营商有深度合作才能够实现的。

（3）轮询机制。这种方式是目前的主流，包括 IOS 采用的也是一样，轮询需要考虑两个问题，一个是服务器负荷问题，另一个是轮询间隔决定推送的及时性，二者是相互矛盾的，轮询间隔短的话对服务器的性能要求就会很高。当然，轮询的间隔可以不是固定的数值，可以通过特定的算法形成一个动态的值，以达到最佳推送结果。

1.5 主要工作内容

本文设计并实现了基于地理位置的社会化购物、分享以及即时优惠券推送的 LBS 系统平台，通过此平台可以将优惠券基于用户的地理位置推送给合适的消费者，用户也可以通过此平台得到更多想要的优惠。论文对 Push 型位置服务应用技术进行了分析，并主要介绍了整个系统的架构和设计思路，主要工作内容如下：

（1）对现有 Push 型技术进行分析，针对 Push 技术和移动终端的特点，提出了移动设备的 Push 机制实现方法。

（2）针对 LBS 服务的对象，找到 Shopylife 需要实现的功能点，并对这些需求进行详细的分析和介绍。

（3）针对分析得出的功能需求，设计出系统的整体架构和开发环境部署，并详细划分各大功能模块为多个子功能模块，使每个模块完成相对独立的功能，然后再进行详细的数据库设计。

（4）通过系统的子功能模块分析，实现 Web 系统，API 系统，移动客户端的开发，并进行系统测试和优化。其中，Web 系统和 API 系统采用 LAMP 架构开发，移动客户端采用 Xcode+Java 语言开发。

1.6 本论文的章节安排

本论文分为八个章节，各章节安排如下：

第一章为绪论，主要介绍了本文的项目背景、项目的概述，然后分析了 Push 技术的特征和基于地理位置的即时信息推送（Push）服务的应用趋势，并对本文的研究内容进行了简要的介绍。

第二章 关键技术介绍，主要介绍了开发 Shopylife 系统使用的主要技术，包括

（LAMP、AJAX、Sphinx 搜索引擎、Python）。

第三章 Shopylife 需求分析，主要介绍了 Shopylife 的项目在开始阶段的需求分析，以及需要实现的具体功能。

第四章 Shopylife 系统设计，介绍了 Shopylife 的整体架构、业务流程、服务器架构、模块设计，以及系统数据库的设计。

第五章 Shopylife 特色功能模块开发与实现，阐述了 Shopylife 的一些特殊功能模块在开发过程中所运用的技术和方法以及运行结果。

第六章 优化与测试，对整个系统的性能进行优化，并运用各种测试方法对整个系统进行功能测试和性能测试，兼容性测试，安全性测试，以及在测试的过程中遇到的问题和解决方法。

第七章 总结和展望，对整个项目期间所做的工作进行总结，并分析了目前的优缺点，展望了下一步系统需要改进的地方。

第二章 关键技术介绍

Shopylife 是一个融合了互联网的 WEB 应用和移动互联网的手机应用为一体的多平台的社会化购物系统。WEB 应用使用了 PHP 作为主要的脚本语言，同时，也使用了 Python 作为服务器的后台执行脚本。

2.1 Shopylife 系统的关键难点

在 Shopylife 中，如何吸引商家在店铺中添加商品和优惠券，以及鼓励用户使用并分享优惠券。光是功能新颖还不行，用户可能用了一段时间就会慢慢失去兴趣，而且这种模式也很容易被抄袭。

在 Shopylife 系统当中，也有一些技术难点，例如需要通过 Oauth 技术来连接其他的社会化账号，以便同步分享数据，在 Facebook，Twitter，Foursquare，Gowalla，Brightkite 和 Linkedin 这几个网站中 brightkite 的 API 一直有些问题。

所以要做到一次签到，分享到全部的 LBS 网站还是有一定困难的。

2.2 相关名词解释

Coupon（优惠券）：是商家为了推出自己的产品，而向用户发送的一种促销券，用户可以在手机上收集该优惠券，并到发布该优惠券的商家那儿得到折扣。

Item（产品或商品）：是指商家在 Shopylife 这个平台上发布的个人店铺内的商品。

Reward（活动）：是指商家为了聚集人缘，让附近的人来参加商店的活动，谁最先完成活动规定的要求，谁就将获得奖励。

Spot（地点）：可以是用户签到的某个小商店，也可以是商家自己添加的地点。

CEO：商家店铺的地主，就是在该店铺签到最多的人。

CFO：这个是 CEO 的好友。

CTO：和 CFO 一样，也是 CEO 的好友，这 3 个称呼没有什么别的意思，就是一个好友互动。手机上的 CEO 可以选择他的朋友来成为 CFO 或 CTO。

2.3 LAMP

LAMP（Linux+Apache+MySQL+PHP）是一组常用来搭建动态网站或者服务器的开源软件，由于该组合是免费的，所以在软件开发的投资成本低，因此受到业界的广泛关注，是目前互联网软件开发最强大的 Web 应用程序平台。

Shopylife 选择了 CentOS 操作系统作为整个系统的运行环境，CentOS（Community ENTerprise Operating System）是 Linux 发行版之一，也是一个开源的 Linux 操作系统。

Apache 是目前最流行的 Web 服务器软件之一，由于其跨平台和安全性而被广泛的使用。Apache 的特点是简单、速度快、性能稳定，所以 Shopylife 选择了 Apache 作为 Web 服务器。

无论是中小型网站，还是大型系统，MySQL 凭借其快速的处理数据的能力，以及低成本和高稳定性被广泛的使用。目前，最新版本为“5.5.21”，Shopylife 选用了这一版本的数据库。

PHP 是英文超级文本预处理语言 Hypertext Preprocessor 的缩写。它是一门跨平台的动态脚本语言，语法简单，使用方便，能很好的支持 MySQL 数据库，且执行效率高，所以被广泛的运用。Shopylife 采用的 PHP 框架是“ThinkPHP”，它是一个简化企业级应用开发和敏捷 WEB 应用开发的开源框架。

2.4 AJAX

Ajax 是指异步 JavaScript 及 XML（Asynchronous JavaScript And XML）。Shopylife 使用了目前普遍流行的 JavaScript 框架 Jquery 来解决 AJAX 的一些浏览器兼容问题。Jquery 是轻量级的 JavaScript 库，不仅兼容各种浏览器（IE 6.0+, FF 1.5+, Safari 2.0+, Opera 9.0+），而且还兼容 CSS3，jQuery 使用户能更方便地处理 HTML documents、events、实现动画效果。

2.5 Sphinx

由于 Shopylife 系统存在大量的地理位置数据，直接搜索建索引的速度很慢，而且搜索的时候需要全表扫描，多个表联合扫描，这样就很难用最快的时间找到搜索结果。一般情况下，都是使用 SQL 语句中的“Like”操作来做数据搜索，这

种方法在数据量比较小的时候效果还行，但是当数据量大到一定程度的时候，这种方法的速度是相当慢的，所以我们需要找到另外一种方法来解决这个问题，那就是建立全文索引，Sphinx 就是一个基于 SQL 的全文检索引擎，它为 MySQL 设计了一个存储引擎插件。我们只需要通过程序语言（如 PHP）来调用 Sphinx 提供给我们的搜索接口就可以很快地完成搜索。因此，Shopylife 将运用 Sphinx 为 MySQL 的存储引擎提供搜索服务，并完成高性能的搜索功能，使用 Sphinx 的时候需要注意的地方是，数据表必须要有数据类型为整型的主键，而且配置的时候需注意对中文的支持。

2.6 Python

Shopylife 使用 Python 编写了一些脚本来实现一些如邮件发送等定时任务以及服务器日常处理。Python 是一种动态脚本语言，它通过解析执行，能够在多种平台下运行。在 Python 中，一切皆是对象，且不需要用大括号来划定程序块，入门简单，由于其也是开源的，因此具有丰富的库，可以实现各种功能。

2.7 本章小结

本章主要介绍了 Shopylife 系统的关键难点、相关名词解释以及整个系统所用到的程序语言和技术架构。后面的章节中，我将对整个 Shopylife 系统的设计与实现具体阐述。

第三章 系统分析

3.1 可行性分析

Shopylife 系统的可行性分析重点解决这个基于地理位置的信息服务推送系统个方面条件，包括技术能否实现，经济效益能否超过开发成本，操作是否可行等问题。

3.1.1 技术可行性分析

采用 Lamp 架构来搭建 Shopylife 的 Web 系统，使用 Mac OS X + Xcode+Java 也能够实现 iPhone App 的开发，通过调用 Web 系统的 API 接口来请求数据交换。所有的物理设备都已经具备，关于这些技术也有相当丰富的学习资料，因此整个系统在技术实现上是完全可以的。

3.1.2 经济可行性分析

本系统商业模式清晰，发展前景很好，已经获得个人风险投资，并且开发成本很低，运行费用也是按流量来算，在前期可能需要增加一些推广和运营方面的成本，但是当积累了一定数量的用户过后，由于具有很好的盈利模式，因此，可以在很短的时间内收回成本。

3.1.3 操作可行性分析

Shopylife 既有基于 B/S 模式的 Web 系统，又有基于移动终端的手机应用，整个系统操作简单，引导性强，能够更方便让用户熟悉使用该系统。

3.2 具体需求分析

Shopylife 系统包括了用户前台系统和商家后台管理系统，以及 API 系统和 iphone 应用。网站前台系统主要展示了用户推送动态、以及附近的场所、活动和优惠券。其主要内容来自于我们的手机客户端，而网站只负责显示。商家后台管

理系统是完全独立于前台系统的，它主要负责统计商家店铺在网站上的一些数据信息，比如他们的店铺被多少人 check-in 了，商品被多少人浏览过等等，有些统计信息时需要付费才能看到的。API 系统提供了手机开发需要的所有接口，通过这些函数便可以了解到整个手机应用的功能需求。其中前台系统和 iphone 应用是面向所有用户的，商家后台管理系统是面向与我们合作的商家，而 API 系统则为 iphone 等智能手机提供内部接口面向的是系统管理员。

因此 Shopylife 系统的用户主要有 3 种，普通用户一般是可以使用系统一些普通的功能，如账户管理，收集优惠券，参加活动，分享、评论动态等。合作商家可以免费使用商家管理后台的一些功能，还有另外一些功能是需要付费才能使用的，如优惠券的详细统计，发布优惠券、活动的数量等。系统管理员可以使用 API 系统进行接口调用，也可以对普通用户和合作商家的不法操作进行控制。

3.2.1 Shopylife 前台系统需求分析

前台系统主要由账户系统、内容展示和搜索功能这三大功能组成，如表 3-1 所示。

表 3-1　Shopylife 前台系统基本功能

账户系统：	内容展示：	搜索功能：
1、用户注册	1、Push 动态	1、地点搜索
2、用户登录、登出	2、附近地点	2、优惠券搜索
3、私信	3、附近优惠券	3、活动搜索
4、好友关注	4、附近活动	
5、社会化账号连接	5、店铺	
7、店铺申请	6、优惠券	
	7、产品	
	8、活动	
	9、会员卡	

账户系统的具体功能分析：

1）注册功能：目前 shopylife 实行了邀请码的注册机制，也就是说用户必须要有一个邀请码才能成为本站的会员。

2）登录、登出：可以用注册邮箱登录，也可以用用户名登录，如果频繁登录的话就需要输入验证码才行。

3）私信：跟站内信一样，需要双方都互相加为好友才能发送消息给对方。

4）关注与取消关注：如果遇到有相同爱好的人，可以选择关注他，当然，如果不喜欢的话也可以取消关注。

5）社会化账号连接：这里主要有 4 个网站可以连接，Foursquare、Twitter、Gowalla、Facebook。

6）店铺申请：店铺申请主要是为商家从普通用户成为商家用户提供的一个申请过程。每一位申请者都需要验证，有两种验证方式，一种是商家提供验证文档，另外一种是商家申请电话验证。上传文档验证需要等待 24 个小时，如果验证成功，我们会将验证码发送到用户的注册邮箱。如果申请电话验证，我们会在 24 小时内拨打用户填写的电话号码并验证用户信息是否属实，若符合条件，用户则会得到一个验证码。用户将验证码输入后就可以认领这个店铺了。

内容展示的具体功能分析：

1）Push 动态：出现在首页，有新的动态立即显示出来，这里的动态可以是来自好友的分享，关注的店铺所发布的优惠券、活动等。Push 动态实现滚动显示，每条动态都可以进行评论和分享。

2）附近地点：这是网站最重要的一个功能，这里的地点一般就是指我们所说的“场所”，网站的所有内容都与“场所”相关。由于绝大多数 LBS 系统都没有统一的场所数据库，比如有个餐馆叫“长城”，在 Foursquare 有，在 Gowalla 上也有，在其它的 LBS 网站上可能同样存在。一个应用程序可以使用很多的定位技术来确定设备的精确坐标，但是坐标只是涉及到地图上的点，我们签到的不是 GPS 坐标，而是某个餐厅或商场。这就需要将我们的坐标连接到场所数据上，场所数据可能涉及大量不同的数据点，但在从最简单的层面上说，场所数据给地理坐标附上了一个名字，以及其他信息等。附近地点主要展示附近 10km 的场所，并分页显示出来。每一页的场所都会这地图上标注出来。

3）附近优惠券、活动：在找出附近 10km 的“场所”之后，分类显示出所有的优惠券以及活动信息。

4）优惠券：其实就是等于广告，包括展示优惠券的各种信息。

5）店铺：展示店铺的商品、优惠信息、活动信息等，用户也可以关注店铺。

6）产品：展示店铺商品的信息，用户可以对该产品做出评价“喜欢”或是“不喜欢”。

7）活动：展示活动的内容，活动时间，活动所属地点，活动的方式。

8）会员卡：用户可以通过扫描与我们合作商家店铺的会员卡的二维码来获取更多的积分。

搜索功能的具体分析：搜索需要输入两个信息，一个是“What”，另一个是“Where”，即基于地点的搜索。还可以选择在具体某个分类下进行搜索，搜索结果以最热或是最新或是最晚来排序。搜索框下面会有一些搜索提示。

3.2.2 Shopylife 商家后台系统需求分析

商家后台管理系统是专门为商家提供的一个后台管理平台。通过该系统，商家可以查看店铺的签到情况，以及有多少人关注，每天被浏览的次数，以及浏览者的年龄、性别等统计。商家还可以发布优惠券、添加商品、发布活动、添加会员卡、编辑店铺信息等。主要功能如下表所示。

表 3-2　Shopylife 商家后台系统基本功能

Coupon：	Reward：	Item：
1、添加优惠券	1、添加活动	1、添加商品
2、删除优惠券	2、删除活动	2、删除商品
3、修改优惠券信息	3、修改活动信息	3、修改商品信息
4、优惠券分享统计	4、活动类型管理	
5、优惠券使用统计	5、活动参与人数统计	
	6、活动分享统计	

Coupon 是优惠券，其实也就是商家发布出来，有需要的人，就可以收藏起来，然后去店里买东西，然后减价。我们逛街的时候，经常看到店铺上面会说减价，但是，到底什么在减价，我们并不知道，有的人就不想浪费时间进去看，到底什么在打折，如果这个商家可以把 Coupon 放上来，那么，手机用户可以用搜到， 也可以按照距离被推送。

添加优惠券：商家填写完 coupon 的内容，然后，选择需要要发布的店铺，例如一个商家有多个店铺的话，这里同一个优惠券可以发布到很多个分店地址下，商家还可以选择要分享的网站，facebook，gowalla 等等，然后这条优惠券信息也会显示在其分享的网站中去。

优惠券分享统计：记录包括 Check-in、Follow、Collection 等对优惠券进行操作的人，并以饼状图的形式显示。

优惠券使用统计：用柱状图表示最近一周该优惠券的使用情况。

其实，Coupon 并没有太多的互动，因为，每个人到店里买，都可以拿到折扣，因为商家本来店里就在打折，我们主要的是，给商家方面，告诉他们可以发广告，有更多的客户会被你吸引来，Coupon 最主要方便的是商家，所以，商家后台审核

Coupon 是不需要的，他们只要知道多少人收藏了，然后多少人到店里来了，他们基本上是可以数到的。

当然为了鼓励用户收集，在 Coupon 信息显示的时候，我们只显示信息，而不显示 Coupon 的条形码（Coupon 条形码就是商家在后台，自己输入一段少于 10 个字节的代码），这个条形码只有在被收集后才会显示，也会被收藏到用户的优惠券收藏夹里面。

Reward 活动就不同了，活动是需要互动的，用户上传一个“符合要求”的照片，在商家审核以后，客户就得到一个奖励提示，这样的话，在这个奖励有效日期前去领，就可以了。

商家可以选择按签到次数或是上传某个特定的图片来做奖励：

1）按照 Check-in 次数，在指定的时间段内谁 Check-in 次数多，谁就是赢家（一段时间内不允许重复 Check-in），后台不需要审核，但需指定有几个赢家，活动结束后发通知给商家以及赢家。

2）上传特定的图片，可以是针对某个景物的图片，店内某个 Logo 的合影之类，总之是这么一种玩法，每个用户只能上传一次。图片活动需审核，但是前期商家依然要指定有多少个赢家，每通过审核一个，则可用奖励数减一。

Reward 活动可以设定开始时间在未来的某个时间段，时间一到才显示到相应的位置或者页面。

添加功能：主要包括优惠券（Coupon）、活动（Reward）、商品（Item），一般商家只能免费发布 10 个优惠券，10 个活动，添加 20 个商品，如果想要发布更多的内容就需要付费了。

1）添加优惠券：必须填写优惠券内容，原价，现价，图片和描述，开始时间，以及二维码。

2）添加活动：包括提供的奖励，选择活动类型，获奖人数限制，开始时间和结束时间。

3）添加商品：需要上传商品图片，填写商品名称和描述。

删除功能：所有删除操作都不做物理删除，而是保存在删除记录里面。

修改功能：所有信息的修改都必须在发布之前，发布过后，该信息就只能删除了。

3.2.3 API 系统需求分析

API 接口系统全部采用 POST 请求，可返回 XML 或者 JSON 格式数据。请求方法名必填，其它参数按 API 参数选填。

主要功能包括用户相关的 API、照片标签 API、地理位置点记录 API、优惠券相关 API、产品相关 API、分类 API、照片相关 API、会员卡 API。

用户相关 API 主要包括以下功能，如下表所示：

表 3-3　用户相关 API 接口功能

1、用户登录	8、粉丝列表	15、修改头像
2、退出登录	9、检查是否有新的好友添加信息	16、获取未读信息的数目
3、检查用户是否已被注册	10、未通过好友列表	17、设置信息为已读
4、关注某人	11、根据用户 id 取用户信息	18、根据 ID 返回信息内容
5、添加好友、关注好友	12、获取当前用户信息	19、获取登录用户的信息列表
6、检查这个用户是否关注过	13、查找好友	20、检查消息发送是否成功
7、用户注册,注册前先检测用户是否注册	14、获取这个用户所有关注（follow）的店铺 id，名字，照片，时间等	21、检查删除消息是否成功

照片标签 API 主要功能，如表 3-4 所示：

表 3-4　照片标签 API 接口功能

1、插入标签链接	4、取消关注标签	7、关注某个标签
2、获取照片标签的距离	5、获取附近的标签	8、获取某个标签下最新的 4 张图片
3、检查标签是否关注过	6、获取某个时间内的标签的所有照片	9、获取某段距离内的标签的所有照片

地理位置点记录 API 主要功能，如表 3-5 所示：

表 3-5　地理位置点记录 API 接口功能

1、根据地点 ID 获取地点信息	5、获取地点附近的所有用户动态	9、获取指定坐标 10km 附近的所有店铺
2、搜索地点	6、收藏店铺	10、获取店铺下面所有的商品
3、判断地点是否收藏过	7、取消收藏店铺	11、获取店铺下面所有的优惠券
4、添加场所	8、记录用户查看店铺的历史	12、将店铺绑定到指定的商家

优惠券相关 API 主要功能，如表 3-6 所示：

表 3-6 优惠券 API 接口功能

1、根据优惠券 ID 获取地点信息	5、获取店铺下的所有优惠券	9、获取指定坐标 10km 附近的所有优惠券
2、搜索优惠券	6、收藏优惠券	10、兑换优惠券
3、判断优惠券是否收藏过	7、取消收藏优惠券	11、添加优惠券
4、获取优惠券的收藏者列表	8、记录用户查看优惠券的历史	12、根据条形码获取优惠券

商品相关 API 主要功能，如表 3-7 所示：

表 3-7 商品相关 API 接口功能

1、添加商品	5、获取店铺下的所有商品	9、根据商品名称获取商品信息
2、根据分类获取商品列表	6、获取指定地点附近的商品	10、记录用户查看商品的历史
4、根据商品名称获取商品的优惠信息	8、根据商品 ID 获取商品信息	12、根据条形码获取商品信息

分类 API 主要功能，如下表所示：

表 3-8 分类相关 API 接口功能

1、获取地点分类 ID 的下一级子分类 ID	3、在某个产品分类下最后追加一个子分类	4、获取产品分类 ID 的下一级子分类 ID
2、添加某个地点分类子分类	4、获取指定地点附近的商品	

照片相关 API 主要功能，如表 3-9 所示：

表 3-9 照片相关 API 接口功能

1、拍照上传	6、获取用户的相片列表	10、记录用户查看照片的历史
2、根据照片 ID 获取信息	7、删除相片	11、获取店铺下面所有优惠券
4、获取店铺下面所有的照片信息	8、获取好友和自己的所有相片信息	12、根据照片标签获取所有属于这个标签的详细信息
5、评论相片	9、获取相片的所有评论	

会员卡相关 API 主要功能，如表 3-10 所示：

表 3-10 会员卡相关 API 接口功能

1、获取运营商列表	6、获取卡片分类列表	10、添加卡片分类
2、根据运营商 ID 获取相关信息	7、根据分类获取所有的卡片列表	11、获取店铺下面所有的优惠券
4、添加运营商	8、删除运营商	12、添加新卡片
5、获取卡片编号	9、卡片搜索	编辑卡片信息

3.2.4 移动手机应用需求分析

基于手机自身的特点，使得我们在做客户端的时候必须得考虑它的两个限制，一个是手机电量，另一个是网络流量。在设计 Push 算法的时候，必须要考虑到这种设计对二者的影响。因此，在满足即时信息推送的情况下，如何最大限度减少手机电量消耗和降低网络流量是我们必须要考虑的问题。下面我们将分析 Shopylife 在手机应用的功能需求。

ShopyLife 手机应用的主要功能包括：Activity（Push 动态）、Point(地点)、Coupon(优惠券)、Card(会员卡)、Camera（包括照相和扫描）等。

其用例关系图如图 3-1 所示：

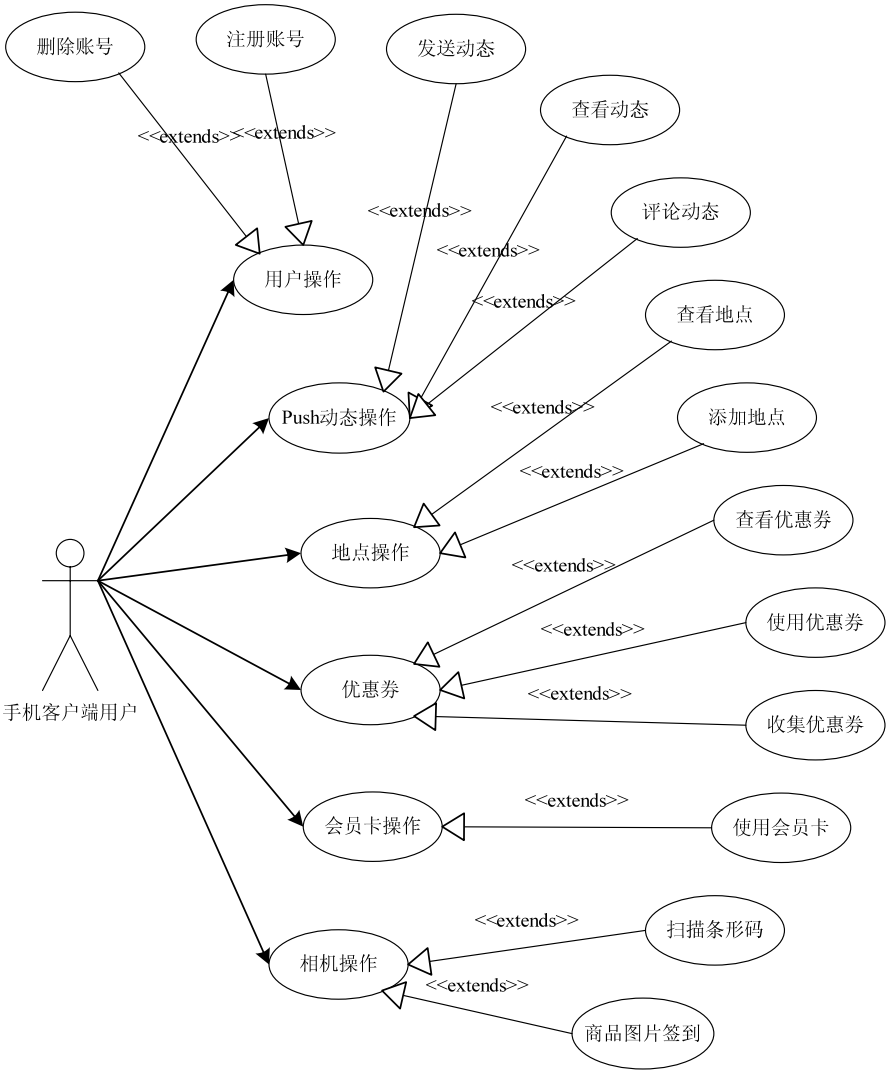

图 3-1 Shopylife 客户端用例图

Push 动态操作：显示 Push 消息，完成 Push 消息的读取，评论、收集等操作。当手机用户解锁屏幕时插入一次轮询，启动通讯录的时候插入次轮询，使用 WIFI 时降低轮询的周期等等操作来让用户感觉到是即时的更新。

Push 服务器一有新的优惠券或活动内容更新，就立即通知移动应用，移动客户端会和服务器保持一段时间的连接，并向服务器发起请求数据，让用户在第一时间收到商家发布的消息，从而保证了优惠券被使用的几率。

即时信息推送的用例关系，见下图 3-2：

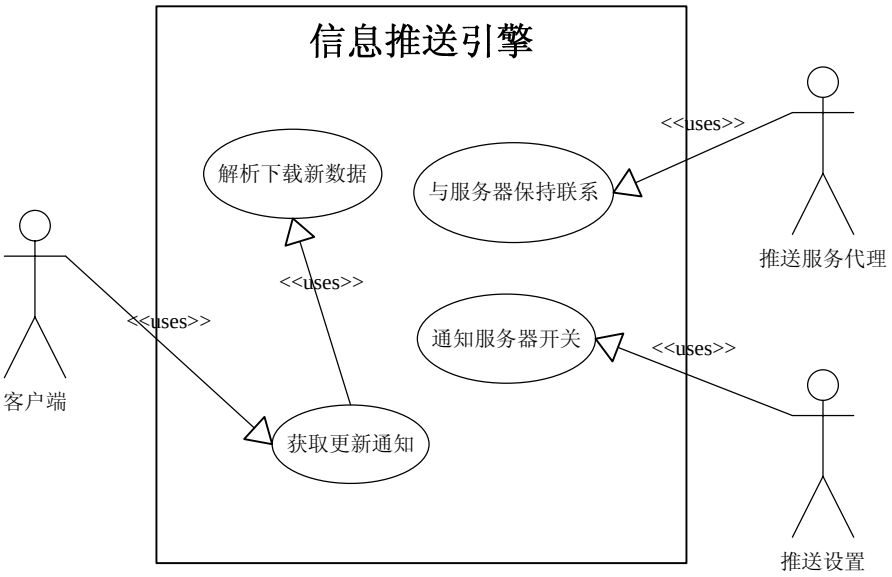

图 3-2 Shopylife 即时信息推送引擎用例图

地点操作：对于商家来说可以为店铺添加商品，优惠券，活动，查看统计信息等，对于普通用户来说，可以“喜欢”店铺的某个商品，也可以对地点进行图片签到，如果签到的地方还没有被添加过，可以添加该地点的一些场所信息。

优惠券操作：商家可以通过手机发布优惠券，用户可以通过手机收藏优惠券，然后将优惠券的二维码出示给商家，商家需要和自己发布的优惠券二维码进行对比，如果匹配便可以给出优惠。

会员卡操作：商家可以添加自己商店的会员卡，普通用户可以通过刷手机上的会员卡积分来获得更多的优惠。

在智能手机上的相机操作，包括以下两个功能：

1）Snapshot Deals：就是打开照相机功能。点击 Snapshot deals，打开的照相机功能，照下来的照片，保存到手机里面，然后出来周边的地址列表，需要选择这

个照片的所在店铺。选择好以后呢，会进入到发布页面，可以选择是不是要分享到其他网站，可以写上自己的一小段话，也可以选择同时分享给好友。

2）Scan Barcode：就是扫描商品上面的条纹码。点击 Scan Barcode，会进入扫描过程，扫描到的数字会推送到服务器，然后由服务器在商品列表里面，寻找是不是有配对的商品，如果有的话就返回商品图片，商品名称，条形码数字。点击完成后，服务器会定位用户，然后寻找这个产品在这个用户周边的店铺里面是否有卖，以及多少价格。

3.3 本章小结

本章首先对 Shopylife 系统进行了可行性分析，主要包括技术可行性，经济可行性和操作可行性。然后对用户前台子系统、商家后台子系统、API 系统、移动手机应用四个子系统进行了具体的功能需求分析，为下一章系统设计做好准备。

第四章 系统设计

4.1 总体设计

根据系统分析结果，Shopylife 系统从结构上应满足：

1）基于浏览器和基于手机客户端进行显示，以方便用户使用。

2）采用 LAMP 技术实现网站系统和 API 系统。

3）采用 MVC 的三层体系结构，分化各个功能组件。

4）采用 Memcache 技术缓存数据。

本系统的体系结构如下图所示。

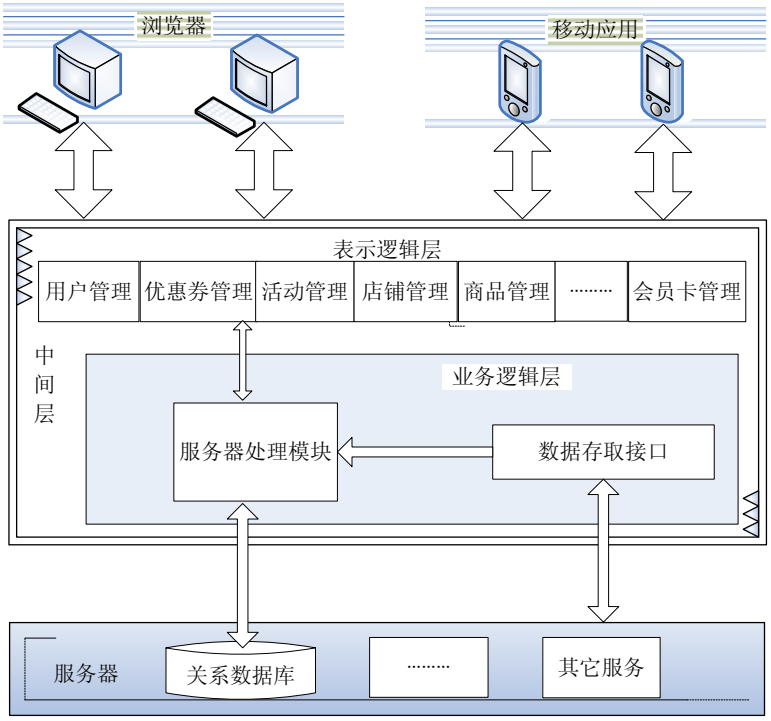

图 4-1 Shopylife 的体系结构

客户层主要是指用户登陆的 Web 浏览器或手机客户端，中间层负责平台得业务逻辑处理和表示逻辑生成，服务器层提供最底层的数据库服务器。

根据功能分解，Shopylife 系统被分解成商家管理子系统、用户管理子系统和手机客户端子系统。根据页面流的设计，商家管理子系统又分为店铺管理、商品

管理、活动管理、会员卡管理等 4 个模块，用户管理子系统又分为用户管理、动态管理、好友管理、标签管理、社会化账号管理，手机客户端子系统又分为 Push 动态管理、签到管理、评论管理、历史记录管理等，如图 4-2 所示。模块之间的关系，如图 4-3 所示。

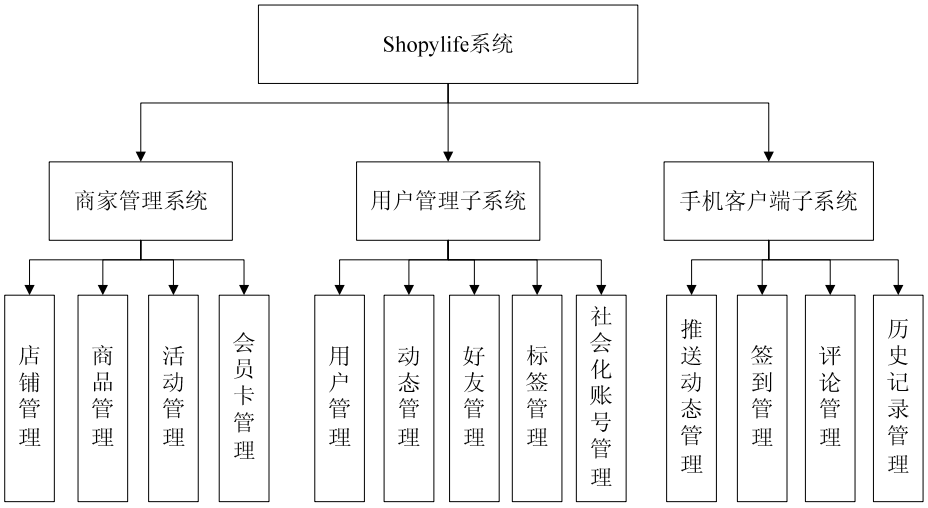

图 4-2 Shopylife 的模块设计

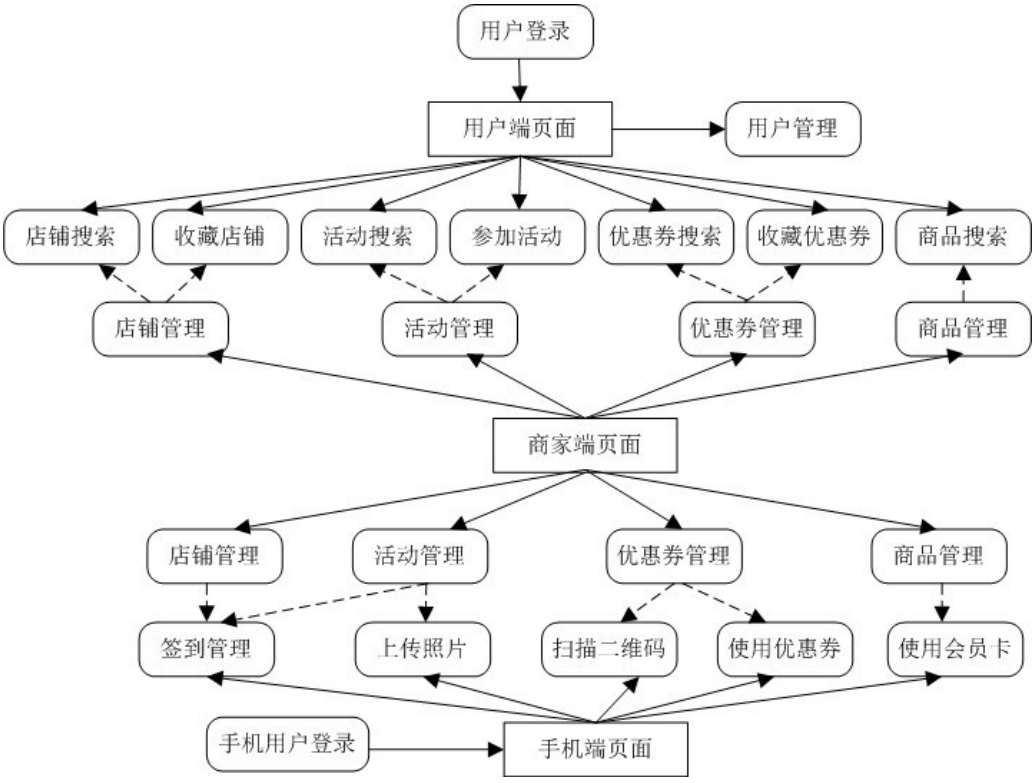

图 4-3 Shopylife 模块之间的关系图

4.2 服务器构架

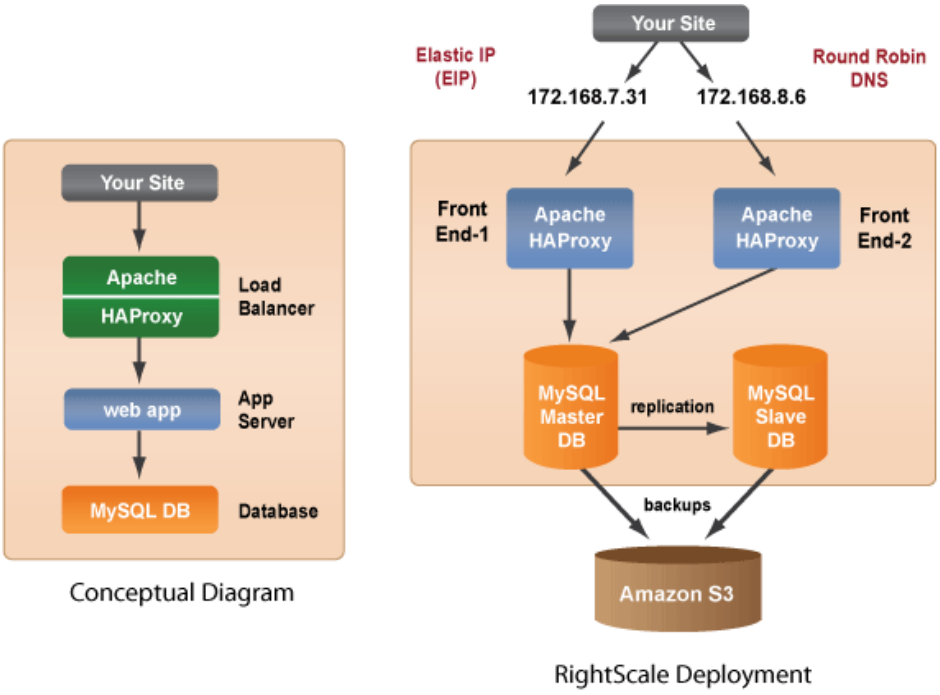

图 4-4 Shopylife 服务器布置

如图 4-4 所示，服务器目前采用 RightScale 的方式来布置，该服务器为核心 PHP 应用程序运行环境。RightScale 是一家提供对多个云平台统一访问的公司，他们最近发布了云管理平台的新版本，其中包含了新的 MultiCloud API、新的增加服务器助手程序以及社区翻译。其中最有意义的功能就是新的 MultiCloud API。到现在为止，RightScale 的 API 1.0 已经支持 Amazon 的云平台，但是他们想要的是通过统一的 API 覆盖所有云平台。API 既支持 XML 也支持 JSON 格式，JSON 是默认格式，并且它是围绕 REST 规范来设计的。

以下是 List Servers 命令的示例：

GET https://my.rightscale.com/api/servers

下面是启动服务器的示例：

POST https://my.rightscale.com/api/servers/11/launch

(1)然后是数据库，为了缓解数据库读取操作压力，数据库引擎采用了 Master/Slave 架构，将写入与读取操作分开，MySQL 主从同步原理是 Master 上提供 Binlog（日志文件），Slave 通过 I/O 线程从 Master 拿取 Binlog，并复制到 Slave

的中继日志中，Slave 通过 SQL 线程从 Slave 的中继日志中读取 Binlog，然后解析到 Slave 中。

(2)然后是图片，这些签到的照片会上传到亚马逊简单存储服务（Amazon S3，Amazon Simple Storage Service）里面。

4.3 Push 服务器推送模型

Shopylife 采用基于 AJAX 长轮询的服务器推送模型。这种长轮询方式下，如果没有新的数据需要传递或者连接超时，服务器端会阻塞客户端的请求，客户端是在 XMLHttpRequest 的状态码等于 4 的时候调用回调方法，然后进行信息处理。这种轻量级的传输机制允许我们通过较小的延迟来传送数据，当客户端处理推送的数据或者连接超时后重新建立连接时，服务器端可能又会有新的数据产生，这些数据会被服务器端保存直到客户端下一次建立连接，客户端会一次性地把当前服务器端所有的最新数据取回。

Push 服务器与移动终端之间的关系如下图 4-5 所示。

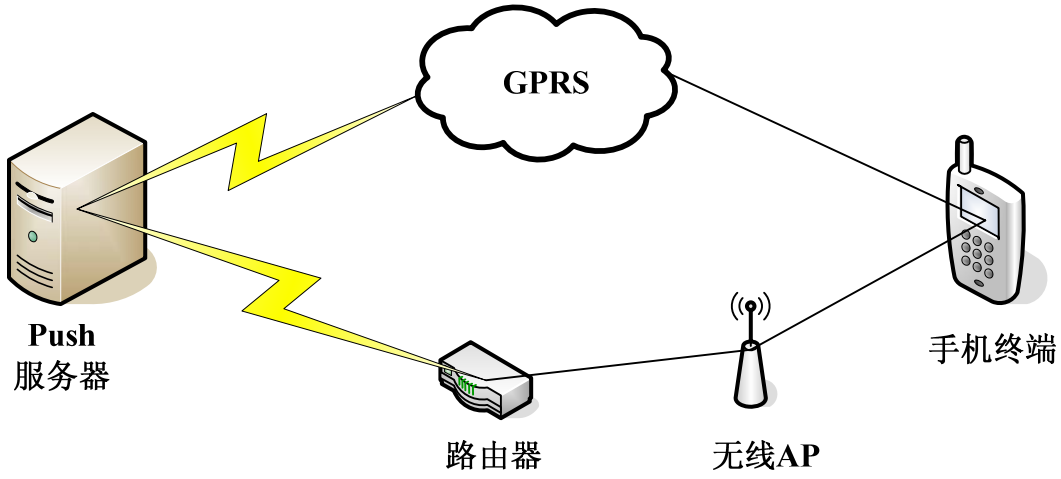

图 4-5 Push 服务框架图

4.4 数据库设计

4.4.1 Shopylife 主要数据表描述

Shopylife 系统总共 22 个表:Member 表、Place 表、Coupon 表、Item 表、Card

表、Place_item 表、Card_code 表、Category_card 表、Place_couon 表、Follower 表、Viewer 表、Comment 表、Card_chain 表、Photo 表、Friend 表、Message 表、Category_item 表、Photo_tag 表、Collect 表、Category_place 表、Tag_link 表、Tag 表。其中各数据表的关系，如下图 4-6 所示。

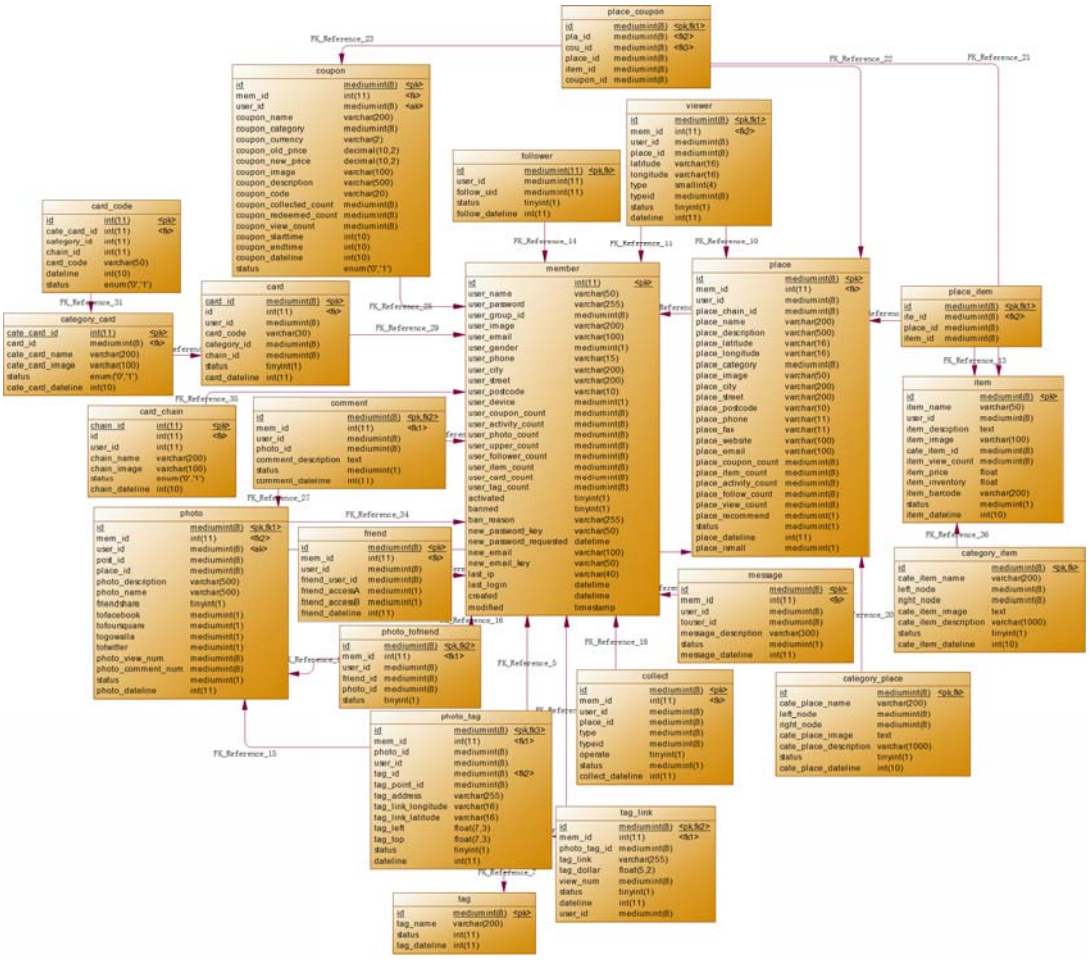

图 4-6 Shopylife 数据表关系图

下面介绍几个比较重要的数据表：

Members 数据表结构，如图 4-7，共 31 个字段，记录用户的各种信息，包括用户名，MD5 加密的密码，群组号，头像，邮箱地址，性别，电话，居住城市，街道，邮编，上游（关注）数，下游（被关注）数，优惠券数量，照片数量，会员卡数量，参加活动数量，账号是否激活，账号是否绑定其它社会化账号，以及最后一次登录的 IP 地址，注册时间，和修改个人信息时间等。

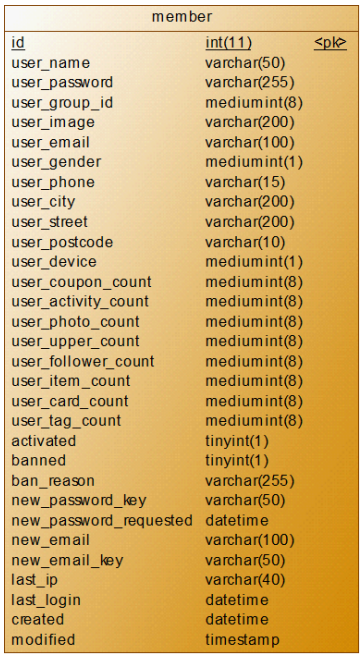

图 4-7 member 表结构

Place 数据表结构，如图 4-8：

地点表主要记录店铺、场所等地点的相关信息，共 25 个字段，包括创建者 ID，该地点的上级地点 ID，地点描述，经度，纬度，地点分类，照片，地点所在的城市，街道，邮编，店铺的链接，店铺的邮箱，店铺总的优惠券数量，产品数量，活动次数，店铺被浏览的次数，以及店铺是否存在等信息。

图 4-8 place 表结构

Coupon 数据表结构，如图 4-9 所示：

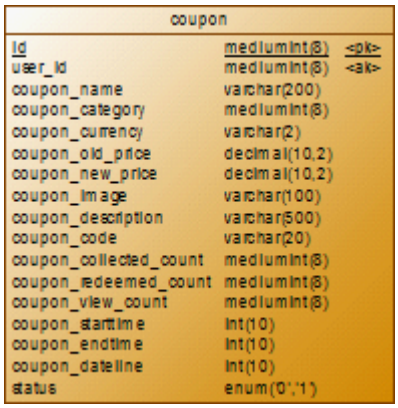

图 4-9 coupon 表结构

优惠券表包括了 17 个字段，主要记录商家 ID，优惠券名称，优惠券分类，原价，现价，优惠券图片，优惠券描述，二维码，开始时间，结束时间，以及一些统计信息，包括被收集的数量，被使用的数量。

Photo 数据表结构，如图 4-10 所示：

照片表包括了 15 个字段，主要记录了用户进行图片签到的一些信息，包括用户 ID，所在地点 ID，照片描述，是否分享给好友，是否同时分享到 Facebook、Foursquare、Gowalla、Twiiter 等网站，是否被删除，签到的时间，被浏览的次数和评论数。

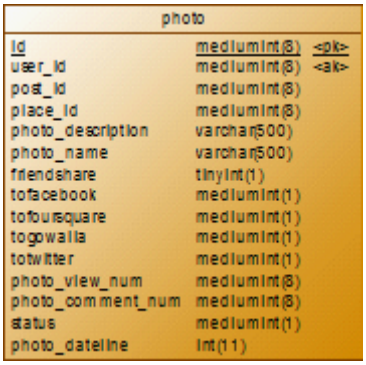

图 4-10 photo 表结构

Item 数据表结构，如图 4-11 所示，商品表总共 12 个字段，记录了店铺中商品的一些详细信息，包括商品名称，商家 ID，商品描述，商品图片，商品所属分类，商品条形码，商品价格，被浏览的次数，剩余商品数量。

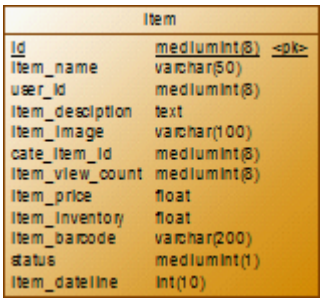

图 4-11 item 表结构

4.4.2 无限分类数据表结构设计

下面我将介绍无限分类的表结构设计。

Category_item 表结构，如图 4-12 所示：

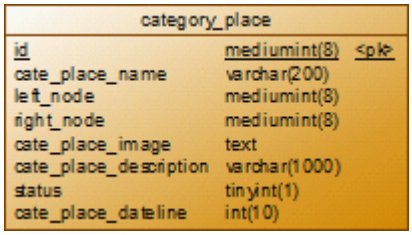

图 4-12 category_item 表结构

商品分类表结构采用前序遍历树结构进行操作，left_node 表示左节点，right_node 表示右节点，cate_Item_name 表示分类名称，cate_Item_image 表示分类图片，cate_Item_description 表示分类描述，cate_Item_dateLine 表示分类创建的时间。

Category_place 表结构，地点分类表和商品分类表的结构基本相似，也是采用树型结构的方式来存储父类和子类的关系。如图 4-13 所示：

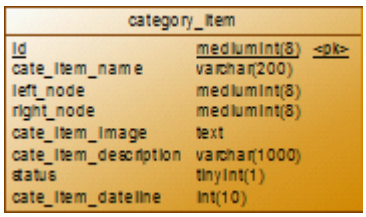

图 4-13 category_place 表结构

采用树型结构可以实现无限分类，父类可以分出它子类，子类又可以分出它的子类，这样一直无限循环下去。

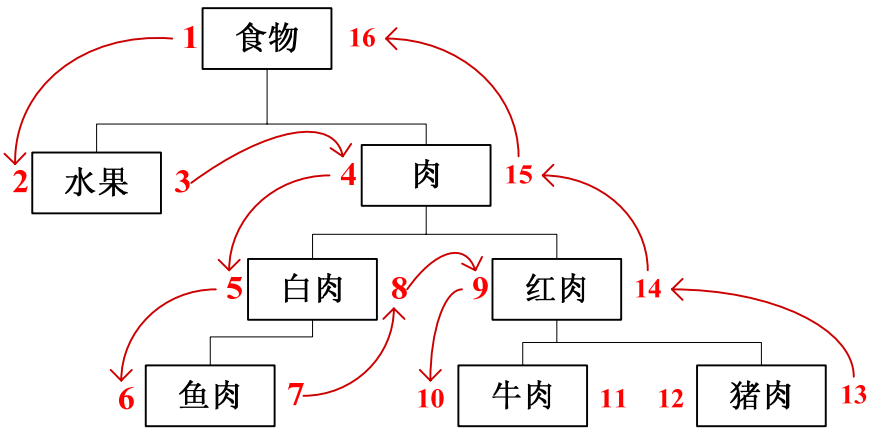

图 4-14 无限分类原理

如上图 4-14 是一颗分类树，从父类到子类用红色的线连接起来了。我们先把这个分类树横向遍历，并且在每个节点的两边边标上序号。

我们把节点旁边的序号叫做左节点序号和右节点序号，由图可见，“食物”的左节点序号是 1，右节点序号是 16，每一个节点序号都暗示了节点之间的关系。“肉”的左节点序号是 4，右节点序号是 15，因此，“肉”是“食物”的子节点。而且，所有左节点大于 4 并且右节点小于 15 的节点，“肉”的子节点。这样，无限分类就通过左节点和右节点保存起来了，如图 4-15。

	id 产品分类id	cate_item_name	left_node	right_node	cate_item_image	cate_item_description	status	cate_item_dateline
	1	食物	1	16		食物	1	1290697248
	2	水果	2	3		水果	1	1290697248
	3	肉	4	15		肉	1	1290697248
	4	白肉	5	8		白色的肉	1	1290697248
	5	鱼肉	6	7		鱼肉	1	1290697248
	6	红肉	9	14		红色的肉	1	1290697248
	7	牛肉	10	11		牛肉	1	1290697248
	8	猪肉	12	13		猪肉	1	1290697248

图 4-15 商品分类数据表

这样的话，就用下面这条 SQL 语句取出所有的分类了：

```
SELECT * FROM category_item WHERE `left_node`>1 and `right_node`<16 ORDER BY left_node;
```

但如何求一个分类下的所有子类又是一个问题，我们发现如果相邻的两条记录第一条的右节点比第二条的右节点大，那么第一条就是第二条的父类，比如“食物”的右节点值是 16 而“水果”的右节点值是 3 那么“食物”是“水果”的父类，但是又要考虑到多级目录。于是我们用一个数组来保存上一条记录的右节点值，再把它和本条记录的右节点值比较大小，如果上一条记录比这一条记录小，则这

条记录不是子类，把它从数组中删除。这样就解决了这个问题。

插入某个分类，首先找到该分类的父类，然后再把所有分类的左节点值和右节点值大于父类的左节点值的节点的左右节点值加 2，最后再插入本节点就行了，可以用一个存储过程来操作：

```
CREATE PROCEDURE `category_item_insert_by_parent`(IN parent_id MEDIUMINT,IN cate_item_name VARCHAR(200), cate_item_description VARCHAR(255) IN cate_item_dateline INT)
BEGIN
DECLARE myLeft MEDIUMINT;
SELECT left_node into myLeft FROM category_item WHERE id= parent_id;
UPDATE category_item SET right_node = right_node + 2 WHERE right_node > myLeft;
UPDATE category_item SET left_node = left_node + 2 WHERE left_node > myLeft;
INSERT INTO category_item(cate_item_name, left_node, right_node, cate_item_description, cate_item_dateline) VALUES(cate_item_name, myLeft + 1, myLeft + 2, l_order, cate_item_description, cate_item_dateline);
commit;
END
```

删除某个分类操作总共需要三步完成：

（1）得到要删除分类的左右节点值。

（2）删除该分类节点的所有子类节点

（3）修改大于要删除分类的右节点值的所有分类。

存储过程如下：

```
CREATE PROCEDURE `category_delete_by_key`(IN id MEDIUMINT)
BEGIN
SELECT @myLeft := left_node, @myRight := right_node, @myWidth := right_node - left_node + 1 FROM category WHERE id = id;
DELETE FROM category_item WHERE left_node BETWEEN @myLeft AND @myRight;
UPDATE category_item SET right_node = right_node - @myWidth WHERE right_node > @myRight;
```

```
UPDATE category_item SET left_node = left_node - @myWidth WHERE left_node > @myRight;
```

4.5 详细设计

4.5.1 用户管理子系统详细设计

用户管理子系统大概包括 20 个页面：首页、注册页面、登录页面、账户设置页面、个人首页、朋友页面、照片列表页面、照片详细页面、产品页面、产品详细页面、地点页面、地点详情页面、地点头衔页面、活动页面、活动详情页面、优惠券页面、优惠券详情页面、地点搜索结果页面、优惠券搜索结果页面、活动搜索结果页面。

1、首页模块设计：

主要展示网站的主题，视频介绍，最及时的动态和正在打折的优惠券。以及一些常见的页面链接。其中的及时动态可以是最新的商品签到，也可以是最新的商店发布的优惠券或者活动，一有更新就滚动显示。

2、注册登录模块设计：

主要需要填写的信息有邮箱地址，全名，用户名，密码，注册语言，验证码等。登录页面中用户也可以使用 Facebook、Twitter 网站的账号进行授权登录。

3、账户设置模块设计：

可以对用户的注册信息进行修改，如忘记密码，可以通过系统发送邮件到注册邮箱，点击我们的密码修改页面，可以对密码进行修改。用户还可以上传自己头像，但是邮箱不能修改，设置性别后也不能修改。

4、个人首页模块设计：

个人首页需要用户登录后才能看到，主要有以下几个功能模块：好友的动态，关注的商家的动态，所有的动态，和所有的图片签到，评论过的动态，站内消息的发送和展示，右边部分是数据统计，主要统计用户的签到数量，优惠券数量，好友数量，发布的照片数量，部分好友展示。

5、朋友模块设计：

可以添加朋友，左边页面展示用户的所有好友，右边页面展示用户的头像，用户名，所在地，签到数量，优惠券的数量，好友的数量，照片的数量。以及连接了哪些社会化账号。

与用户相关的类图如下图 4-16 所示：

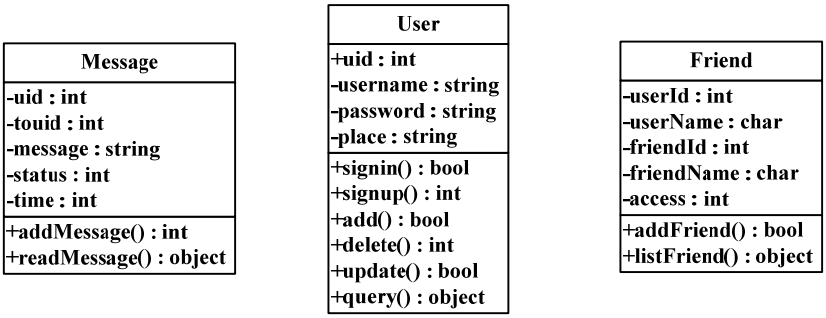

图 4-16 用户相关类图

6、照片列表模块设计：

列出当前登录用户签到的所有的照片，可进行分页操作，每一页展示 25 个照片。右边模块同样展示用户的个人统计信息。

7、照片详细页面设计：

详细页面模块需展示这张照片的详细信息，包括添加者信息，添加时间，照片的描述，照片的评论等信息。登录用户还可以对该照片进行评论。

8、产品模块设计：

即商品模块设计，商品页面主要展示了店铺内所有商品列表，商品信息包括商品名称，商品所在地点，喜欢的人数，不喜欢的人数，该商品还剩余的数量。用户可以对所有商品按照分类过滤，也可以对某个商品标注为喜欢或者不喜欢。

9、产品详细页面设计：

该模块展示产品的详细信息，以及用户对该商品进行签到的所有记录，用户也可以评论这些记录，右边模块会展示该商品总的签到数量，和优惠券。

10、地点模块设计：

该模块展示附近店铺的信息通过一个地图直观的显示离自己所在的地方有多远。

11、地点详情页面设计：

该模块展示地点的详细信息，包括的功能模块有，关注店铺，申请为该店铺的商家，展示店铺的签到数，优惠券数，好友数，以及该店铺的所有签到记录。右边模块记录整个商家的签到数量，优惠券数量，商品数量，照片数量，以及访问该店铺的用户。

12、地点头衔页面设计：

该模块按签到次数最多的用户来展示的。右边模块除了展示一般信息外，还展示访问最多的店铺。

13、地点搜索结果页面设计：

地点搜索结果页，主要搜索包含该关键字的地点名称或地点描述。然后将搜索结果标记在 Google 地图中，其余部分给地图列表显示页面一样。

与地点相关的类图，如下图 6-17 所示：

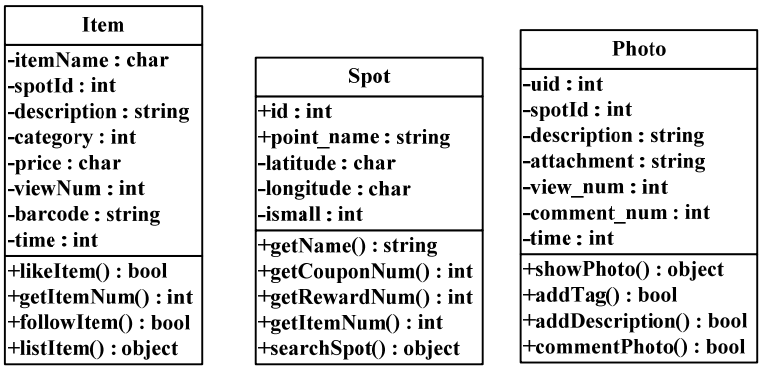

图 4-17 地点相关类图

14、活动模块设计：

该模块用来展示附近的所有活动，每个活动参加的人数，并需求进行分类筛选。

15、活动详情页面设计：

该模块主要展示活动的具体信息，包括剩余时间、活动方式、活动的奖励，活动的具体地点等。

16、优惠券页面设计：

该模块展示附近的优惠券列表，每条优惠券显示多少人收集、多少人使用、剩余的优惠期限，并能根据其分类进行过滤筛选。

17、优惠券详情页面设计：

该模块主要展示优惠券的具体信息。其中有一个醒目的优惠券图片，和大字号的价格，以及多少折扣，多少人收集了，多少人使用了，这几些数据并排显示。然后是一个时间倒计时，表示该优惠券还剩余多长时间。下面会介绍优惠券的详细内容。右边模块会显示优惠券的地图信息，和已经收集了该优惠券的用户头像。

18、优惠券搜索结果页面：

优惠券搜索结果页面，主要搜索在指定地点包含该关键字的优惠券，并像优惠券列表一样展示出来。

19、活动搜索结果页面：

活动搜索结果页面，和优惠券搜索结果页面一样，也是搜索在指定地点的包好该关键字的活动。

与优惠券，活动相关的类图，如下图 6-18 所示：

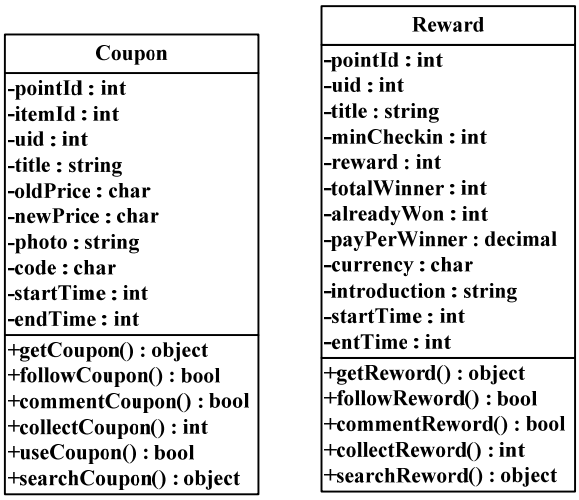

图 4-18 优惠券、活动相关类图

4.5.2 商家后台管理子系统详细设计

商家后台管理子系统大概包括 8 个页面：商家后台首页、签到统计页面、优惠券统计页面、活动统计页面、商品统计页面、添加优惠券页面、添加活动页面、添加商品页面，相关类图如下图 4-19 所示。

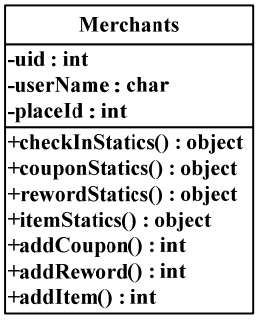

图 4-19 优惠券、活动相关类图

1、商家后台首页：

商家后台首页主要展示商家还有多少可以发布的优惠券，活动数。以及有多少个关注者，以及一些添加按钮。然后会展示一些最近的店铺浏览统计，和浏览者

的身份统计。然后会展示目前的关注着头像信息，和该店铺的分享信息。

2、签到统计页面：

签到统计页面展示所有在该店铺进行签到以及进行社会化分享的用户信息。包括用户头像，用户名，签到的时间，签到的次数，分享的网站。

3、优惠券统计页面：

优惠券统计页面展示店铺内所有的优惠券信息，包括优惠券图像，优惠券名称，优惠开始的时间，优惠券被多少用户收集，剩余的时间，优惠结束时间，并提供一个编辑按钮。

4、活动统计页面：

活动统计页面展示了活动的图片，活动名称，活动开始时间，有多少用户参与，活动剩余时间，活动结束时间，并为每条活动提供一个编辑按钮。

5、商品统计页面：

商品统计页面展示了商品的图片，商品名称，商品被浏览的次数，喜欢这个商品的用户的数量，不喜欢该商品的用户的数量，并为每一条商品信息提供一个编辑按钮。

然后是添加页面，添加操作的时序图如下图 4-20 所示：

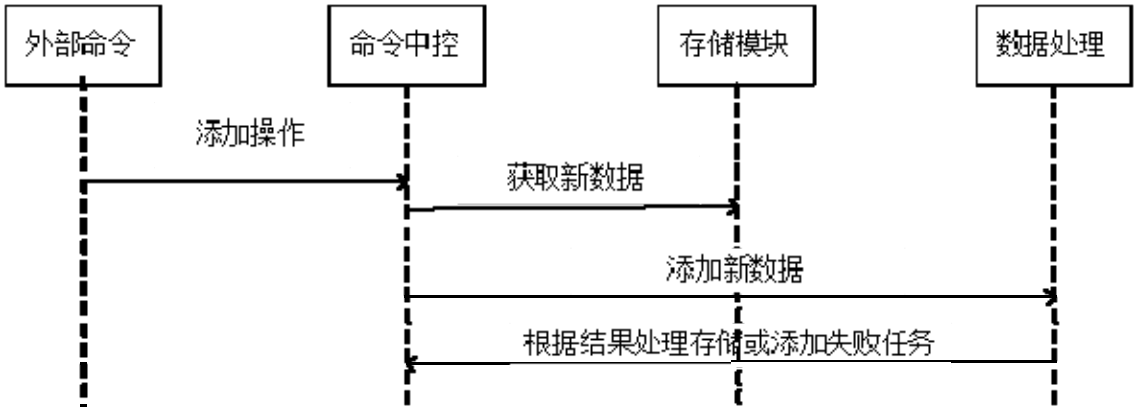

图 4-20 添加操作序列图

6、添加优惠券页面：

添加优惠券页面提供了商家输入优惠券信息的界面，包括优惠信息的文本框，原价、现价的输入框，图片的上传框，图片描述文本框，优惠券开始、结束时间的选择框，以及优惠券生成二维码的字符串输入框。

7、添加活动页面：

添加活动页面提供了商家输入活动信息的界面，和添加优惠券的界面大致相

同，包括活动信息的文本框，活动方式的单选框，以及需满足的签到次数选择框，以及活动奖品输入框，添加活动图片的按钮，图片描述的文本框，活动的开始时间和结束时间的选择框。

8、添加商品页面

添加商品页面提供了商家添加商品信息的界面，包括商品信息的文本框，商品价格的输入框，商品图片的上传按钮，商品图片描述的文本框。

4.5.3 API 子系统详细设计

Shopylife 的 API 系统用 WebService 的方式为手机提供服务，接口函数遵循 REST 风格，使用 XML 或 JSON 为数据交换格式。

主要接口函数设计及参数说明：

1、用户相关

User.signin：用户登录，User.signout：退出登录，User.signup：用户注册，User.addfriendbyid：添加好友，User.getuserbyid：获取用户信息，User.updateavatar：更新用户头像。

2、地点相关

Place.getbyid：返回地点信息，Place.search：根据条件返回搜索结果，参数均为可选，但必须有一个条件才能进行搜索，Place.add：添加地点，Place.getactivity：返回地点下面的所有用户动态，Place.getcoupon：返回地点下面所有的优惠券，Place.getitem：返回地点下面所有的商品，Place.nearby：返回指定坐标附近的所有地点，Place.view：记录用户浏览地点历史。

3、优惠券相关

Coupon.add：添加优惠券，Coupon.search：搜索优惠券，Coupon.colletor：返回 coupon 的收藏者列表，Coupon.nearby：返回指定坐标附近的所有 coupon（可进行分类筛选），Coupon.getbybarcode：返回该二维码的优惠券信息。

4、商品相关

Item.add：添加一个商品，Item.getbybarcode：根据 barcode 获取商品的信息，Item.getbycateid：根据分类 ID 获取商品列表，Item.view：记录用户查看商品的历史。

5、照片相关

Attachment.add：拍照上传，同时可以为照片设置四个标签（图片放置目录原

图：/attachments/iphone/large/，缩略图：/attachments/iphone/thumb/），Attachment.getbyid：根据照片的 ID 编号获取相关照片的详细信息，Attachment.getbytagname：根据照片的标签获取所有属于这个标签的所有信息，Attachment.view：记录用户查看照片的历史，Attachment.getcomment：获取照片的评论，Attachment.remove：删除照片，Attachment.comment：评论照片。

6、标签相关

Tag.addlink：插入一条标签的链接，Tag.follow：设为感兴趣标签。

7、分类相关

Category.placegetchild：获取地点分类 ID 的下一级子分类 ID，Category.itemaddnode：在某个产品分类下最后追加一个子节点，Category.placeaddnode：添加某个地点分类子节点，Category.itemgetchild：获取产品分类 ID 的下一级子分类 ID。

8、会员卡相关

Card.chain.list：获取运营商列表，Card.chain.get：获取运营商详细信息，Card.chain.add：添加运营商，Card.chain.remove：删除运营商，Card.get.code：获取卡片编号，Card.edit：编辑卡片信息。

4.6 本章小结

本章介绍了 Shopylife 的整体架构、业务流程、服务器架构、模块设计，以及系统数据库的设计和各个子系统的详细设计。下一章将会根据此系统设计进行编码实现。

第五章 特色功能开发与实现

根据以上的需求分析和系统设计，本文实现了 Shopylife 系统的所有功能，开发工作涉及到多种技术和多种平台，系统的开发采用敏捷开发的形式，一边开发一边测试，最终实现了一个功能完善且符合用户需求的系统。

在整个系统中采用了 MVC 三层架构体系，M（模型）层负责与数据库的交互，C（控制器）层负责处理一些逻辑，V（视图）层负责前台页面展示。下面将介绍本系统的特色功能的开发以及主要功能的最终实现结果图。

5.1 检测用户位置

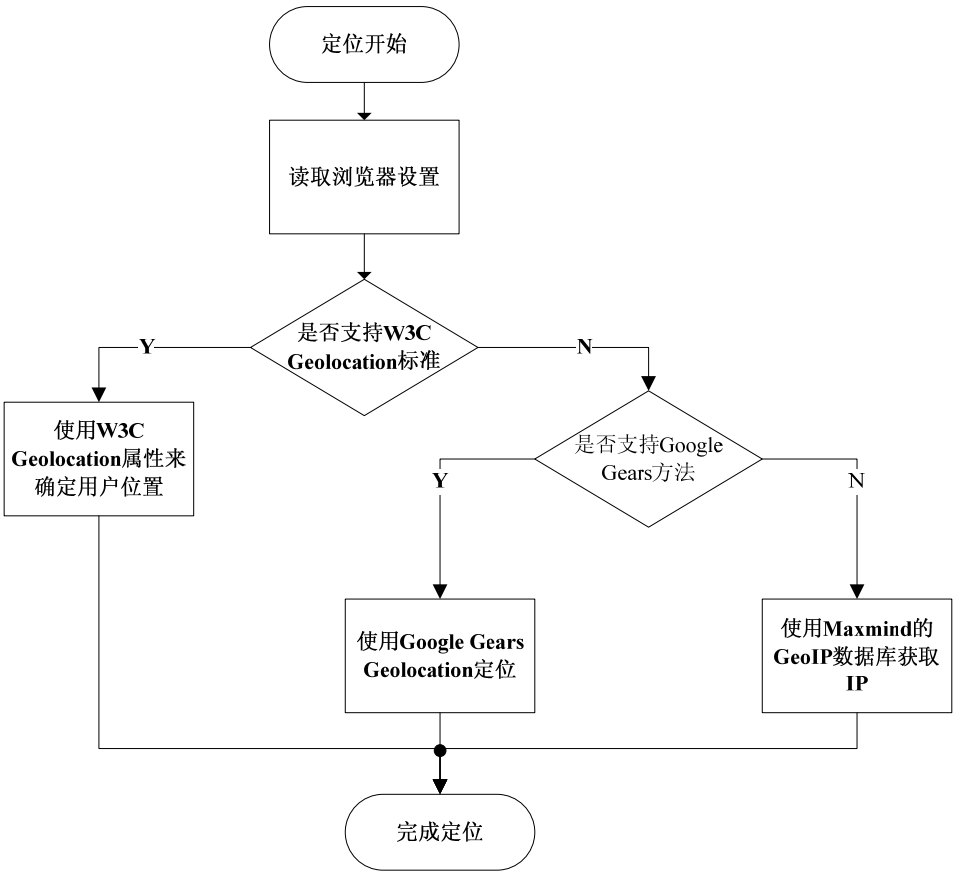

图 5-1 浏览器定位流程图

如上图 5-1 所示，目前，在 Web 中可以用以下 3 中方式来定位。

（1）W3C Geolocation 标准。它是 W3C 在 HTML5 中的一部分，可以通过它来执行地理定位。

（2）使用 Google Gears Geolocation API。由于支持 W3C 标准，某些带 Google Gears 的浏览器可以选择此方法。

（3）使用 IP 地址定位。这种方式是最简单的，但是定位不是很精确。

由于使用 IP 地址不能精确的定位，因此我们先使用能够精确定位的 W3C 方法。Google GeolocationAPI 是 Google 的综合定位接口，该接口使用 HTTP 协议，交互数据为 Json 语法，包括基站定位，Wi-Fi 定位，GPS 定位都可以通过 Gears GeolocationAPI 来查询。我们先用 W3C navigator.geolocation 属性来定位，然后再用 Google Gears，如果两种方法都不行，再利用 Maxmind 的 GeoIP 数据库来获取 IP。代码如下，运行结果如图 5-2 所示：

```
var initialLocation;//初始位置
var beijing = new google.maps.LatLng(40,116);//北京的经度纬度
var chengdu = new google.maps.LatIng(104,30);//成都的经度纬度
var browserSupportFlag =    new Boolean(); //浏览器是否支持 W3C Geolocation 标准
function initialize() {
  var myOptions = {
    zoom: 6, //地图显示大小
    mapTypeId: google.maps.MapTypeId.ROADMAP //显示的地图类型
  };
  var map = new google.maps.Map(document.getElementById("map_canvas"),
myOptions); //新建一个地图
  //使用第一种方式定位
  if(navigator.geolocation) {
    browserSupportFlag = true; //设置浏览器支持该方法
    navigator.geolocation.getCurrentPosition(function(position) {
      initialLocation = new google.maps.LatLng(position.coords.latitude,
position.coords.longitude); //获取当前的位置
      map.setCenter(initialLocation); //设定地图的中心
    }, function() {
      handleNoGeolocation(browserSupportFlag);
    });
```

```
  //使用 Google Gears Geolocation 定位
  } else if (google.gears) {
    browserSupportFlag = true; //设置浏览器支持该方法
    var geo = google.gears.factory.create('beta.geolocation');
    geo.getCurrentPosition(function(position) {
      initialLocation = new google.maps.LatLng(position.latitude, position.longitude);
      map.setCenter(initialLocation); //设定地图的中心
    }, function() {
      handleNoGeoLocation(browserSupportFlag);
    });
  //使用 IP 定位
  } else {
    browserSupportFlag = false;
    handleNoGeolocation(browserSupportFlag);
        var marker = new google.maps.Marker({
                                    position:new google.maps.LatLng(lat,lon),
                                    map:map,
                                    draggable:true,
                                    title:'You are around here!',
                                });
        google.maps.event.addListener(marker,'dragend',function(){changePoint(mark
er.getPosition());});
  }
  function handleNoGeolocation(errorFlag) {
    if (errorFlag == true) {
      alert("Geolocation service failed.");
      initialLocation = chengdu;
    } else {
      alert("Your browser doesn't support geolocation. We've placed you in beijing.");
      initialLocation = beijing;
    }
    map.setCenter(initialLocation);
```

```
}}
```

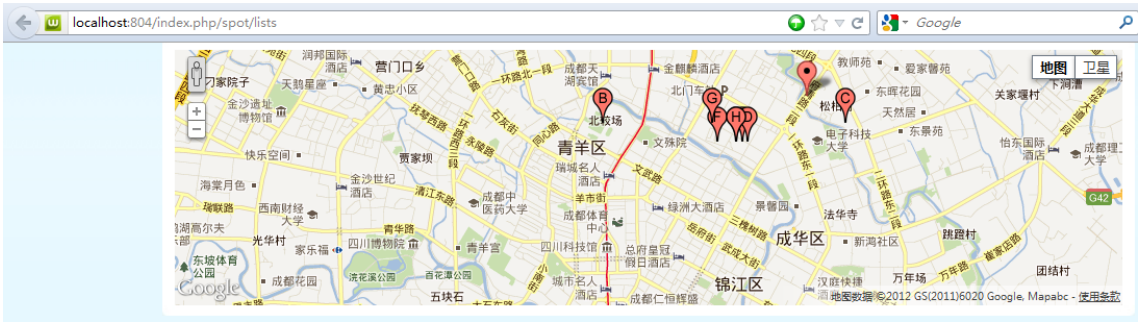

图 5-2 使用浏览器定位

5.2 推送技术实现

在本系统中，我们采用了第一种基于 AJAX 长轮询的方式来实现，轮询不一定是定时定周期的轮询，可以和特定的算法，操作，结合。在一些应用的情况下，可以让人感觉到是及时的更新，这并不难。我们可以通过掌控网络联接，捕获系统消息来控制轮询，比如，屏幕解锁时插入一次轮询，启动通讯录的时候插入次轮询，使用 WIFI 时降低轮询的周期等等。其工作流程如下图 5-3 所示：

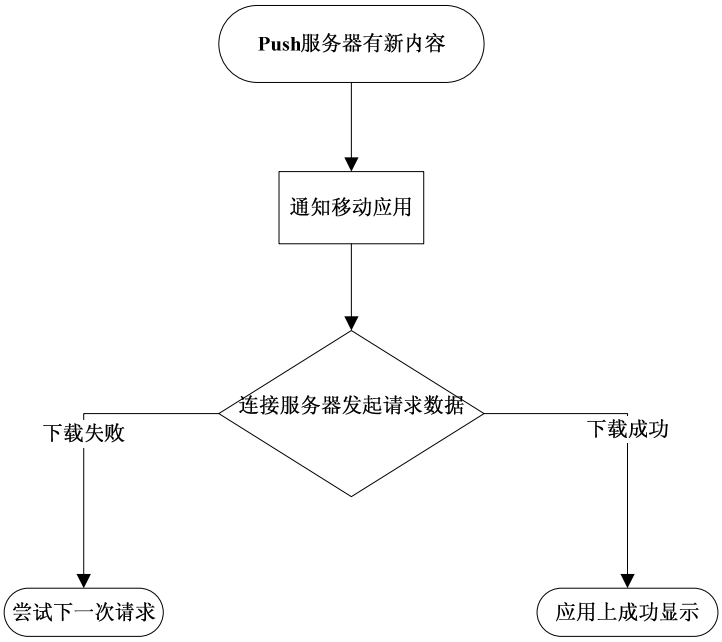

图 5-3 Shopylife 即时信息推送工作流程

我们设定了一个最小的轮询间隔（5 秒），也就是说不能无休止的添加轮询密度，这个数字一般感觉是用户在此系统和使用环境 2-3 次正常单次操作响应过程的时间，这样对于一般的用户体验来说会比较能够接受。再设定一个基本轮询周期（15 分钟），然后可以开始设定一推的规则，用来插入轮询，降低或升高轮询周期。例如解锁屏幕插入一次轮询，手机处在使用状态，轮询周期降到 3 分钟一次，一分钟内发生过 3 次高密度推送的，暂时降低轮询周期到 30 秒等等。只要不违背最小轮询间隔就可以，这些规则需要很长一段时间根据应用的实际情况进行优化，以最大程度提升用户体验，新的规则可以加入，旧的规则可以去掉，规则的参数可以动态的调整。

下面我将用 PHP+Jquery 来简单模拟一下这种轮询模型，代码如下。

服务器端：

```
<?php
$filename = '/data.txt';
$msg = isset($_GET['msg']) ? $_GET['msg'] : '';
if ($msg != '') {
file_put_contents($filename,$msg);
die();
}
$lastmodif = isset($_GET['timestamp']) ? $_GET['timestamp'] : 0;
$currentmodif = filemtime($filename);
while ($currentmodif <= $lastmodif){//检查文件是否被修改
usleep(50000);//轮询间隔
clearstatcache();
$currentmodif = filemtime($filename);
}
$response = array();
$response['msg'] = file_get_contents($filename);
$response['timestamp'] = $currentmodif;
echo json_encode($response);
flush();
?>
```

客户端代码：

```
<script>
    var timestamp = 0;
    var url = 'push.php';
    var error = false;
    function connect(){
        $.ajax({
            data : {'timestamp' : timestamp},
            url : url,
            type : 'get',
            timeout : 0,
            success : function(response){
                var data = eval('('+response+')');
                error = false;
                timestamp = data.timestamp;
                $("#content").append('<div>' + data.msg + '</div>');
            },
            error : function(){
                error = true;
                setTimeout(function(){ connect();}, 5000);
            },
            complete : function(){
                if (error)
              //如果连接出现错误，每 5 分钟进行重连
                    setTimeout(function(){connect();}, 5000);
                else
                    connect();
            }
        })
    }
    function send(msg){
        $.ajax({
            data : {'msg' : msg},
```

```
                type : 'get',
                url : url
            })
        }
        $(document).ready(function(){
            connect();
        })
</script>
```

运行结果，如下图 5-4 所示，该页面会一直等待服务器推送信息，我们通过手动修改“data.txt”文件，让服务器检测到有更新，客户端就会收到更新的内容。也可以通过在客户端提交信息到服务器端来修改“data.txt”文件，同时，在同一浏览器打开另外一个相同的页面可以看到显示的内容与该页面完全一样，而且两个页面可以相互对话。该结果说明以上代码实现了基于 AJAX 长轮询的消息推送。

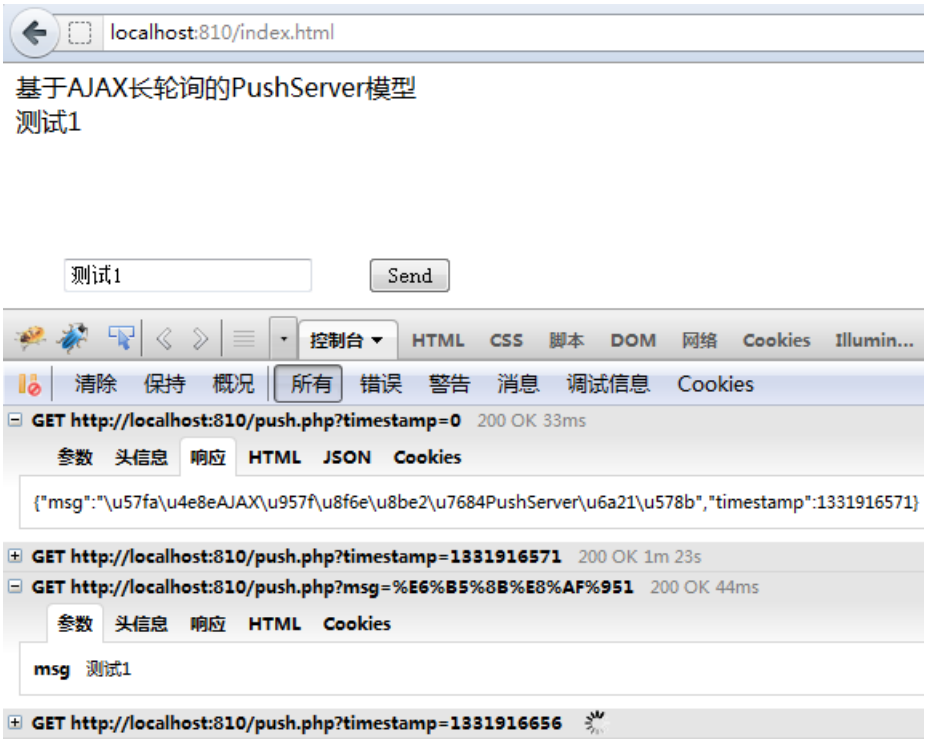

图 5-4 长轮询机制运行结果

当然，一般情况下 Web 服务器允许脚本最大执行时间默认是 30 秒，所以要让程序正常运行，我们可以每隔 30 秒发送一次心跳包，让客户端与服务器保持长连

接。

5.2.1 COMET 关键类

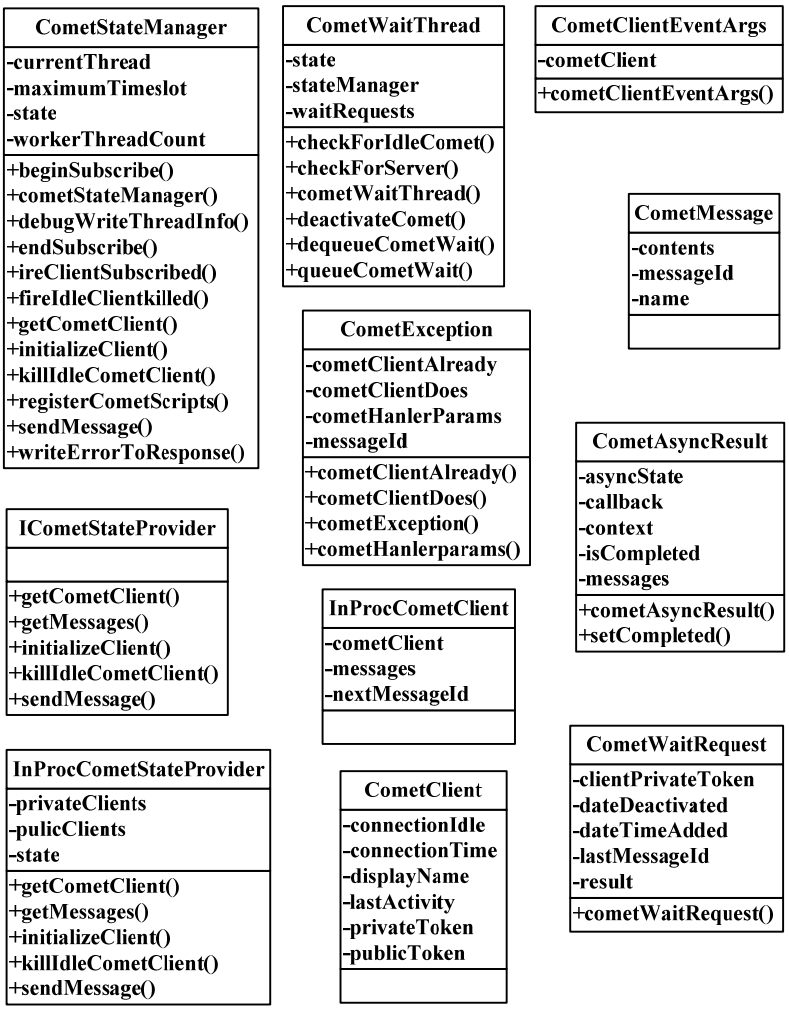

图 5-5 COMET 基类类图

上图 5-5 是推送（Push）功能基类类图，其中关键类说明如下：

1.实体类中，CometClient 类记录了一些服务信息，CometMessage 类为消息传输的主体，InProcCometStateProvider 类继承了 ICometStateProvider 并实现了消息操作思路方法，同时将消息保存在内存中。

2.接口类 ICometStateProvider 定义了一系列接口来提供对 CometClient 实体和 CometMessage 操作。

3.事件类 CometClientEventHandler 负责广播和监听客户端事件。

4.线程相关类中，CometWaitThread 类负责 Push 服务器端线程池等待信息请求

并提供超时等操作，CometWaitRequest 类负责 Push 服务器端监听客户端请求，CometAsyncResult 类负责异步请求管理。

5.主体类 CometStateManager 主要管理线程池管理线程和用户连接消息转发等系列操作工厂类。

5.3 二维码生成

商家添加优惠券的同时，需要为其提供优惠的商品一串字符，这些字符可以是商店的网址、联系信息、电话号码，也可以是任意可识别的字符。添加成功后，系统会自动生成一种可读性的二维条码，可以通过手机或者其他设备扫描识别其中所记载的数据，然后商家通过对比从而给出相应的优惠。

Shopylife 采用矩阵式二维码来记载数据。具有代表性的矩阵式二维条码有 Code One、Maxi Code、QR Code、Data Matrix 等。下面我将实现 QR Code 在本系统中的应用。

通过调用 Google Chart Tools / Image Charts 的 API，我们可以很方便的生成 QR Code。调用方式也很简单，只要向 http://chart.apis.google.com/chart 传入适合的参数就可以了，代码如下：

```
“qrcode.php”
<?php
class qrcode{
private $data;

//生成网址 URL 的二维码
public function link($url){
if (preg_match('/^http:\/\//', $url) || preg_match('/^https:\/\//', $url)){
$this->data = $url;
}else{//网址需要有协议
$this->data = "<a href="http://".$url">http://".$url</a>;
}
}

//创建书签形式网址的二维码
```

```
public function bookmark($title, $url){
$this->data = "MEBKM:TITLE:".$title.";URL:".$url.";;";
}

//创建 E-mail 地址的二维码
public function email_address($email){
$this->data = "<a href="mailto:".$email">MAILTO:".$email</a>;
}

//创建电话号码的二维码
public function phone_number($phone){
$this->data = "TEL:".$phone;
}

//创建短信的二维码
public function sms($phone, $text){
$this->data = "SMSTO:".$phone.":".$text;
}

//创建彩信的二维码
public function mms($phone, $text){
$this->data = "MMSTO:".$phone.":".$text;
}

//创建个人名片的二维码
public function contact_info($name, $address, $phone, $email){
$this->data =
"MECARD:N:".$name.";ADR:".$address.";TEL:".$phone.";EMAIL:".$email.";;";
}

//创建地理信息的二维码
public function geo($lat, $lon, $height){
```

```
$this->data = "GEO:".$lat.",".$lon.",".$height;
}

//获取二维码图片，默认为 150 像素
public function get_image($size = 150, $EC_level = 'L', $margin = '0'){
$ch = curl_init();
$this->data = urlencode($this->data);
curl_setopt($ch, CURLOPT_URL, 'http://chart.apis.google.com/chart');
curl_setopt($ch, CURLOPT_POST, true);
curl_setopt($ch, CURLOPT_POSTFIELDS,
'chs='.$size.'x'.$size.'&cht=qr&chld='.$EC_level.'|'.$margin.'&chl='.$this->data);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
curl_setopt($ch, CURLOPT_HEADER, false);
curl_setopt($ch, CURLOPT_TIMEOUT, 30);
$response = curl_exec($ch);
curl_close($ch);
return $response;
}

//获取二维码图片路径
public function get_link($size = 150, $EC_level = 'L', $margin = '0'){
$this->data = urlencode($this->data);
return
'http://chart.apis.google.com/chart?chs='.$size.'x'.$size.'&cht=qr&chld='.$EC_level.'|'.$
margin.'&chl='.$this->data;
}
}
?>
```

使用方法如下

```
<?php
include("qrcode.php");
$qr = new qrcode();
```

```
//书签形式网址
$title = "购生活";
$url = "http://www.shopylife.com/";
$qr->link($title,$url);
//获取二维码图片 URL
echo "<img src='".$qr->get_link()."'>";
?>
```

生成的二维码如下图 5-6 所示：

图 5-6 Shopylife 网址二维码

5.4 API 系统的实现

Shopylife 的 API 系统实现了以 WebService 的形式为手机提供服务，接口函数遵循 REST 风格，使用 XML 或 JSON 为数据交换格式，如图 5-7 所示，点击 API 链接左边会自动显示需要填写的参数的输入框。

图 5-7 Shopylife API 测试工具

5.5 系统主要功能的实现

5.5.1 消息动态模块

消息动态模块分为未登录动态和登录动态。未登录的动态展示了全网站内用户的签到和分享信息，一有新的消息会向下滚动展示。如图 5-8 所示。

图 5-8 Shopylife 未登录首页动态

登录后的动态只展示了自己好友和关注店铺的动态信息，并且可以分享到其它网站中去，如图 5-9 所示。

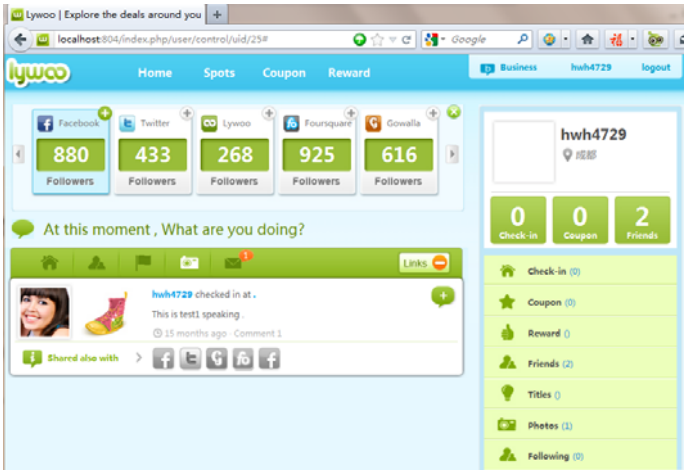

图 5-9 登录后首页动态

手机应用中的动态功能，主要调用接口“attachment.getactivity”如图 5-10 所

示：

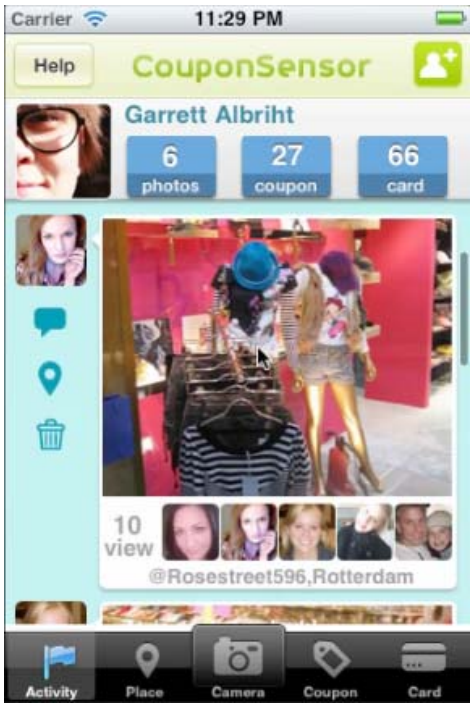

图 5-10 登录手机后首页动态

5.5.2 搜索功能模块

搜索功能模块要实现基于地点的搜索，如果不填地点则默认全部搜索，搜索结果可以按照分类筛选，也可以按照时间、热度排序。运行界面如图 5-11 所示：

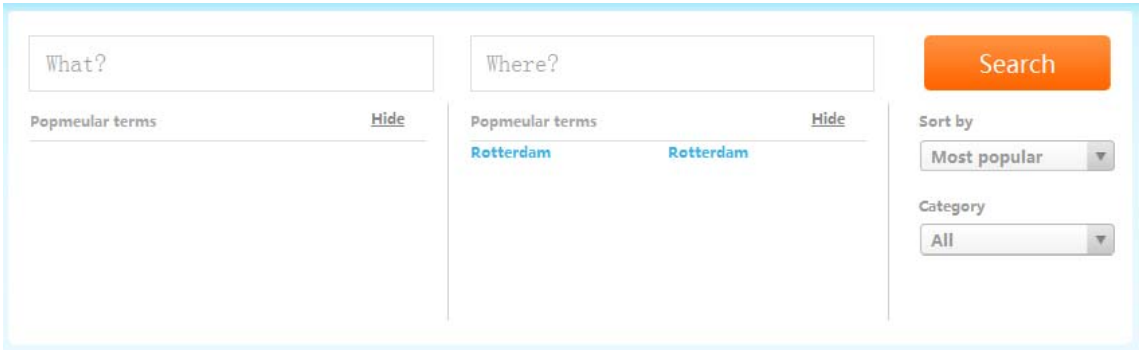

图 5-11 搜索功能模块

搜索功能主要采用 Sphinx 搜索引擎建立索引，其核心代码如下：

```
<?php
```

```
$cl = new SphinxClient ();
    $cl->SetServer ( '*.*.*.*', 9312);//*.*.*.*为服务器 IP 地址
        $cl->SetConnectTimeout ( 3 );//设置超时时间为 3 秒
        $cl->SetArrayResult ( true );//设置结果返回格式，匹配项以普通数组返回，包括匹配项的全部信息（含文档 ID）
        $cl->SetMatchMode ( SPH_MATCH_ANY);//匹配查询词中的任意一个
        $res = $cl->Query ( $data['keyword'], "*" );//查询

        if(empty($res['matches'])==true){//如果没有查询结果，则返回空数组
            $data['total'] = 0;
            $data['news'] = array();
        }else{//如果查到结果，则将关键词标为红色
            foreach($res['matches'] as $key=>$value){
                $news_id[] = $value['id'];
            }
            foreach($res['words'] as $key=>$v){
                $words[] = $key;
                $rewords[] = '<span style="color:red">'.$key.'</span>';
            }
        }
?>
```

手机上的搜索模块主要调用了“place.search”，“card.chain.search”，“coupon.search”这三个接口。

5.5.3 优惠券模块

1、前台页面

如下图 5-12 所示，前台展示模块向普通用户展示了优惠券的名称、描述、现价、原价、折扣，收藏数量，使用数量，剩余时间，以及相关店铺的一些信息，而且告诉了用户收集优惠券只需要三步，第一步，下载我们的移动客户端；第二步，搜索该优惠券，然后“collect”；第三步，出示给商家然后得到折扣。右边部分给出了优惠券所在地点的地图信息，和已经收集了该优惠券的用户头像信息。

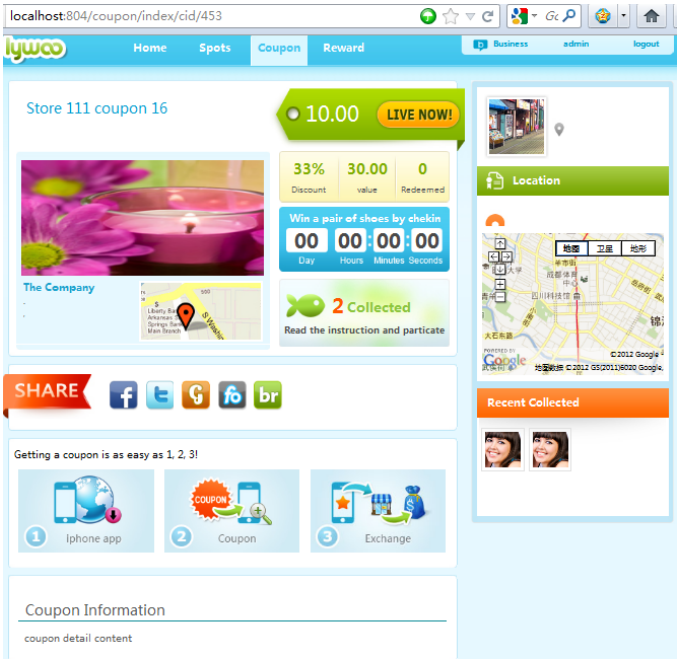

图 5-12 优惠券前台展示模块

手机上的优惠券的页面展示如图 5-13 所示。

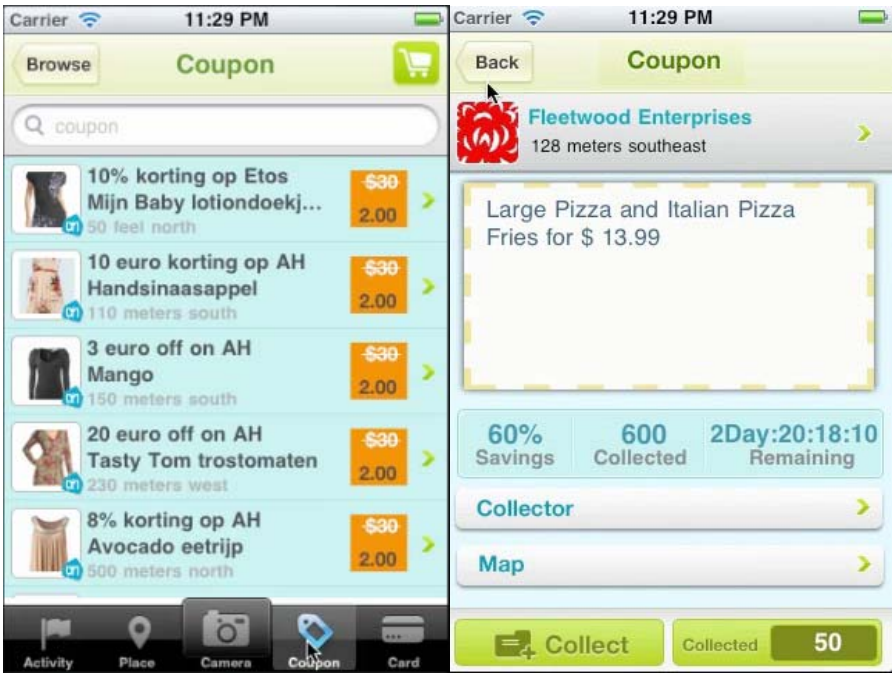

图 5-13 优惠券手机展示页面

2、商家后台添加优惠券

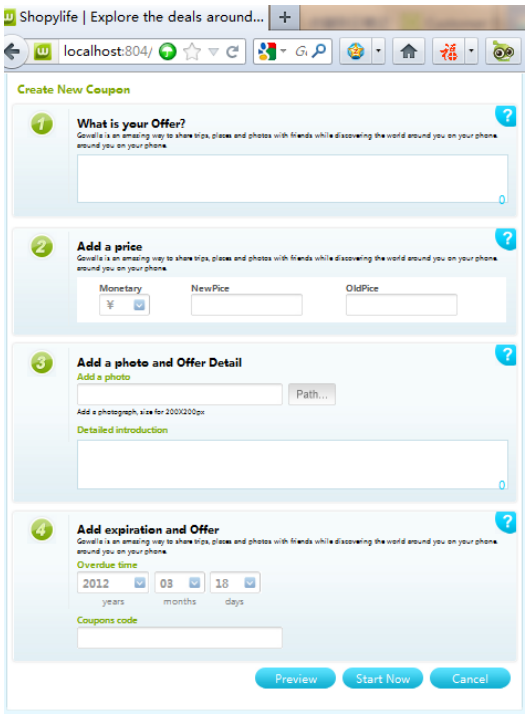

图 5-14 优惠券后台添加

实现了优惠券商家后台添加、编辑、删除，以及被收集和使用的数据统计。添加优惠券页面如上图 5-14 所示，商家可以为该优惠券上传不超过 2 兆的图片，和少于 300 字的图片描述，且商家填写的现价不能高于原价，开始时间不能是过去的时间，这些规则都通过前台 JS 和控制层验证后才插入数据库。

商家后台系统还实现了优惠券的一些统计信息，主要使用了一个轻量级的图表框架 JSChart 来实现一些柱状图，饼图，曲线图等。通过传入一个 XML 数据就可以实现各种统计，代码如下：

```
<script type="text/javascript">
$(function(){
var xml;
var myChart = new JSChart('graph', 'line');
myChart.setDataXML("/business/share");//输出相关统计的 XML 格式数据
myChart.draw();
});
</script>
```

显示结果如下图 5-15 所示。

图 5-15 优惠券统计

5.5.4 用户管理子系统的实现

1、注册页面

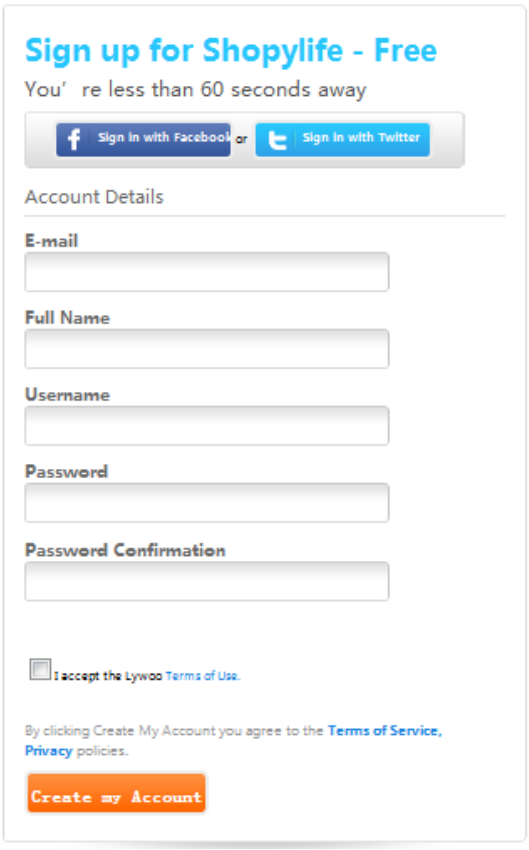

图 5-16 注册页面

2、登录页面

登录窗口界面，如图 5-17 所示，除了用本网站邮箱账号登录外，还可以用

Facebook 账号和 Twitter 账号登录。

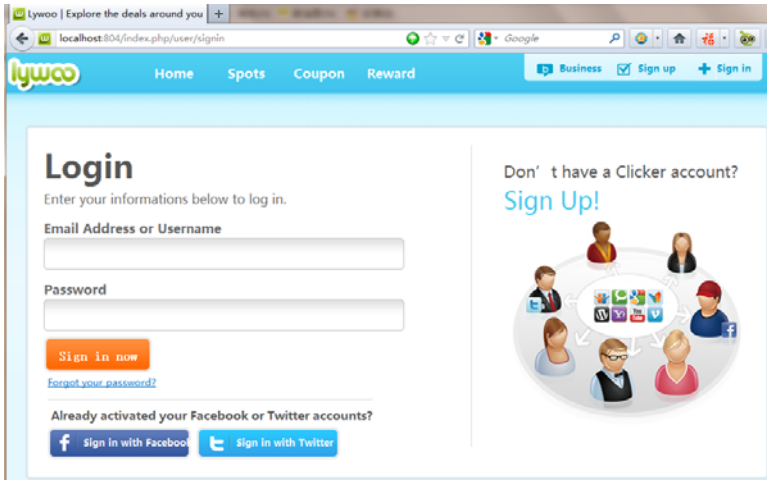

图 5-17 登录页面

3、站内消息

发送站内消息窗口如图 5-18 所示，该窗口以弹出层的方式实现，点击好友下拉框可以选择好友，然后可以输入消息。点击发送，好友就可以收到刚刚发送的消息了。

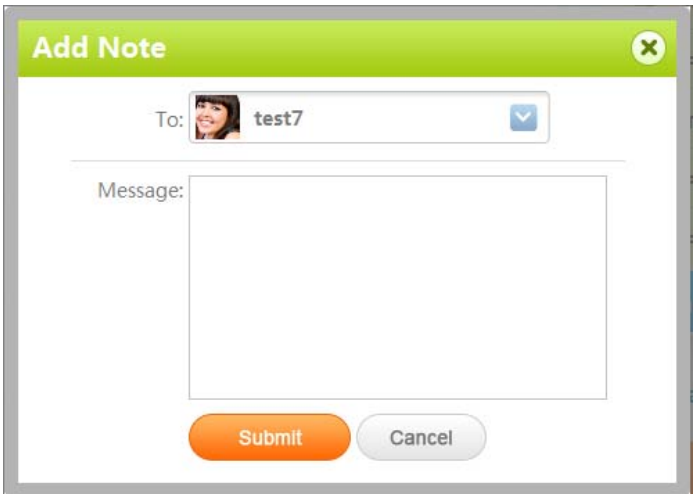

图 5-18 站内消息

4、评论动态

用户可以评论被推送的动态（包括好友签到的商品，分享的优惠券，活动等），评论界面如下图 5-19 所示，该评论是通过 AJAX 的方式实现，点击“cancel”按钮，评论框会自动消失，再点击右上角的绿色按钮，评论框又会出现在这条动态

的下面。

图 5-19 评论动态

手机的评论窗口如下图 5-20 所示：

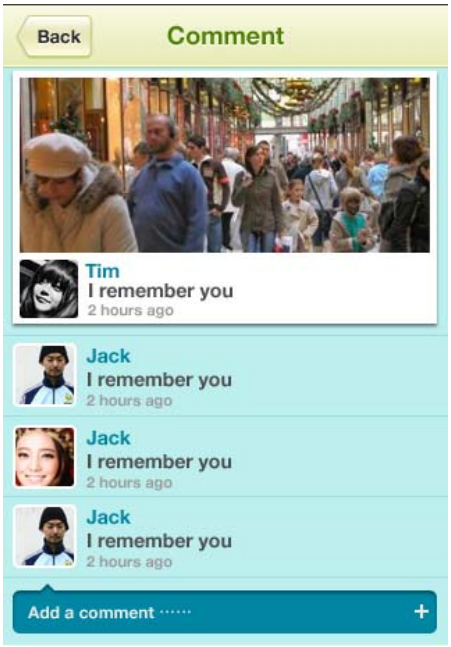

图 5-20 客户端动态评论

5、好友管理

好友管理模块展示了好友的个人信息，如好友头像，好友昵称，好友所在地点。如图 5-21 所示，点击好友的头像可以进入好友的个人主页，进入好友的个人主页可以看到好友的图片签到数量，收集的优惠券数量，以及他的好友数量。还可以看到好友签到的图片以及其他信息。

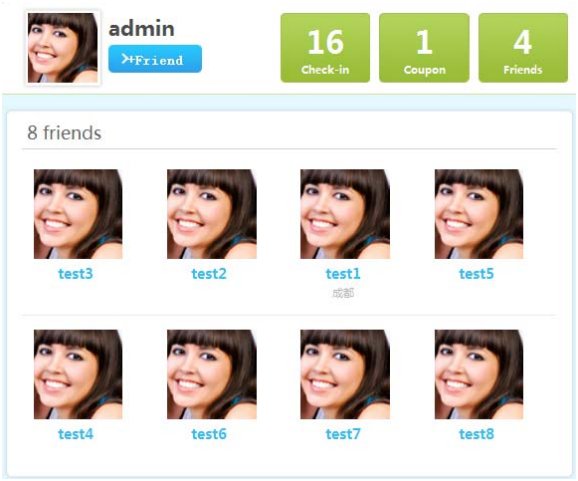

图 5-21 好友管理页面

6、头衔管理

头衔管理模块实现了激励用户参与店铺的商品图片签到中来，签到次数最多的用户将成为该店铺的 CEO 头衔，CEO 可以将 CFO 和 CTO 的头衔授予给他的朋友。如下图 5-22 所示：

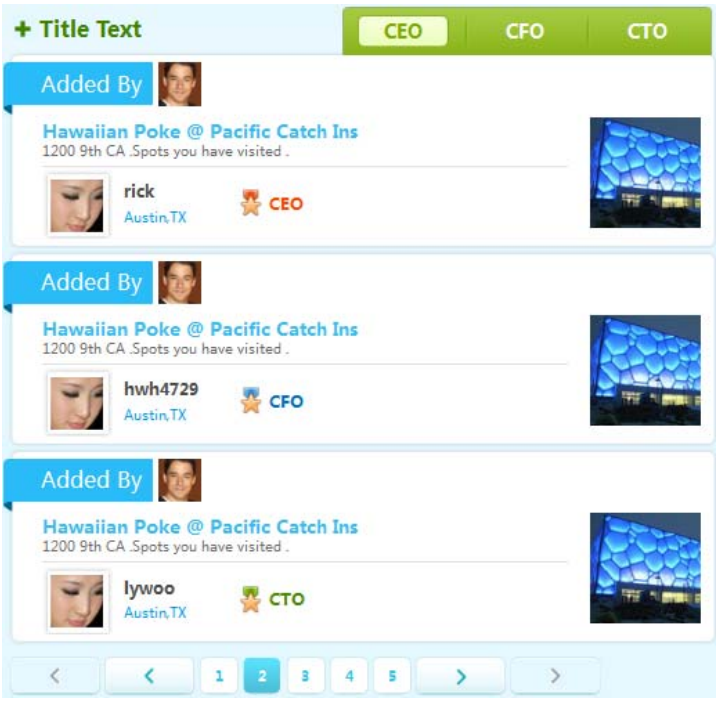

图 5-22 头衔管理页面

7、商家店铺申请

用户管理模块还实现了普通用户成为商家用户的申请过程，即店铺申请过程。如图 5-23 所示，点击 Claim Business 按钮会出现一个店铺申请的弹出层，用户可以按照提示分步骤申请。

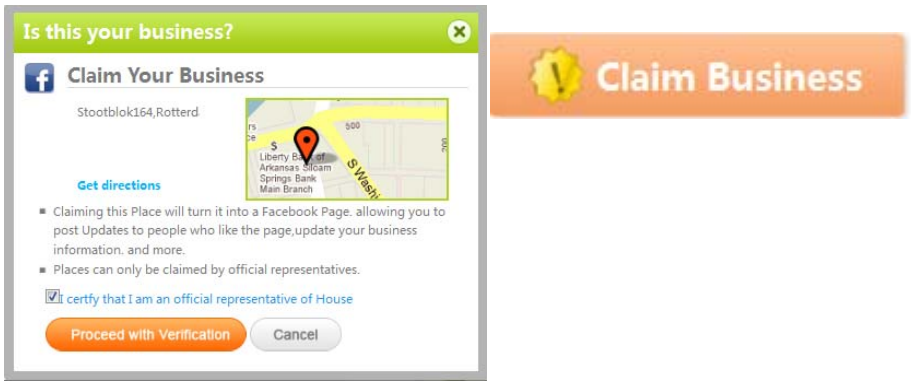

图 5-23 申请店铺过程

8、附近地点展示

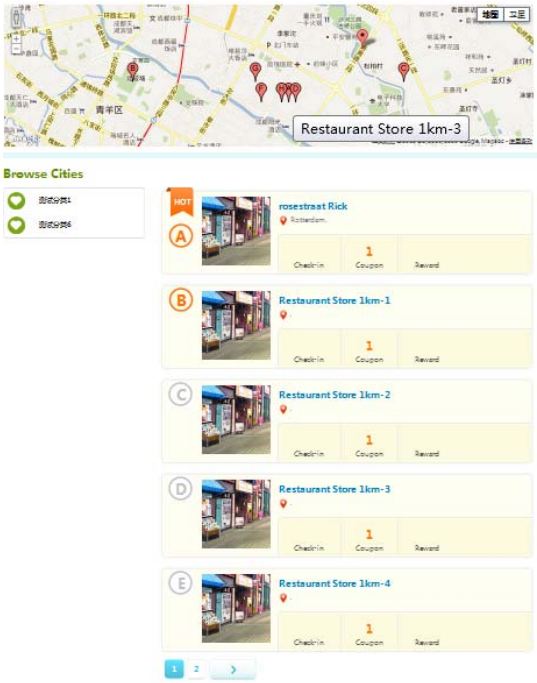

图 5-24 附近地点展示

如图 5-24 所示，当浏览器定位后会显示附近 10km 的店铺信息。每一个店铺会在地图上标记出来，每一个字母代表一个地点，鼠标滑到地图上的字母标记时，会显示该标记下的店铺名称。其中红色的 A、B 地点表示热门的店铺。点击左边的

分类链接，还可以只显示当前分类的店铺。

9、店铺商品展示

如图 5-25 所示，店铺商品展示该店铺下的所有商品列表，可以通过分类进行过滤。点击上方的蓝色下拉框可以选择商品的分类，鼠标滑过分类时会出现当前分类的子分类，选择分类会对该店铺下的商品进行过滤。点击绿色的向上小手表示喜欢该商品的人数，蓝色的向下的小手表示不喜欢该商品的人数，点击小手可以喜欢或者是不喜欢该商品。

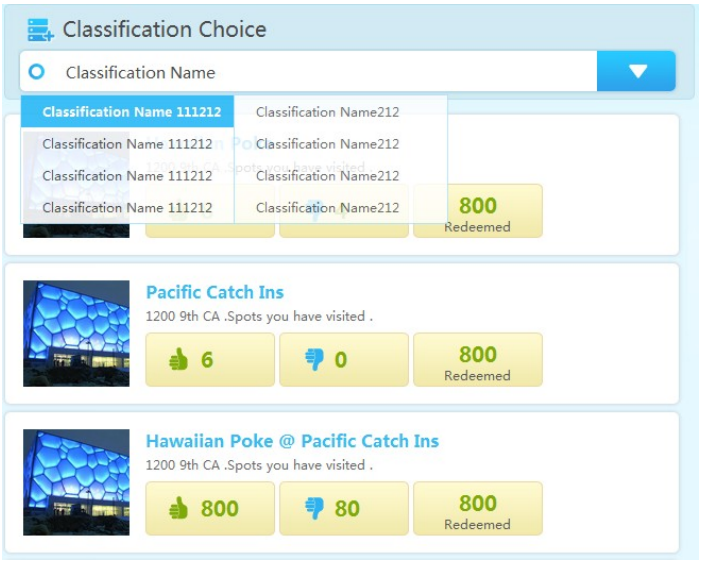

图 5-25 店铺的商品展示

5.5.5 商家后台管理子系统的实现

1、商家后台首页

如下图 5-26 所示，商家后台首页的运行界面如下，导航时 5 个灰色的图标，分别表示首页，签到，优惠券，活动，商品。然后是商家的一些个人信息，可以看到这个商家还剩下 3 个优惠券和 1 个活动可以发布，旁边是一些添加按钮，下面是一些社会化分享信息的统计，然后用一些图片统计了最近 5 天内用户浏览店铺的情况，以及浏览者的性别、年龄统计。

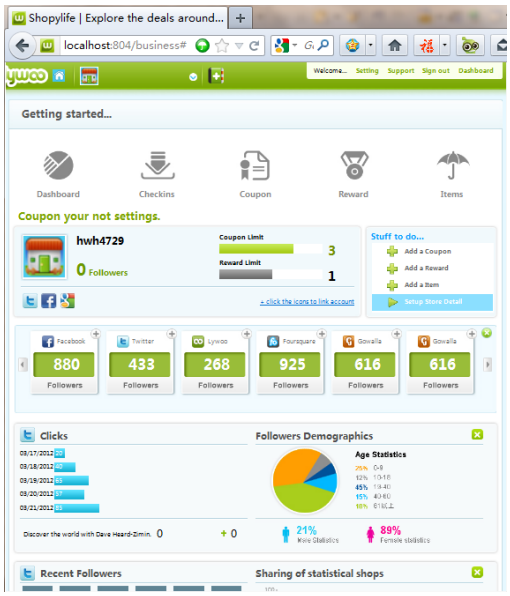

图 5-26 商家后台首页

2、商家后台签到统计页面

商家后台签到统计运行页面，如图 5-27 所示，每页显示 10 条签到数据。点击下面的选择框可以选择任意页数。

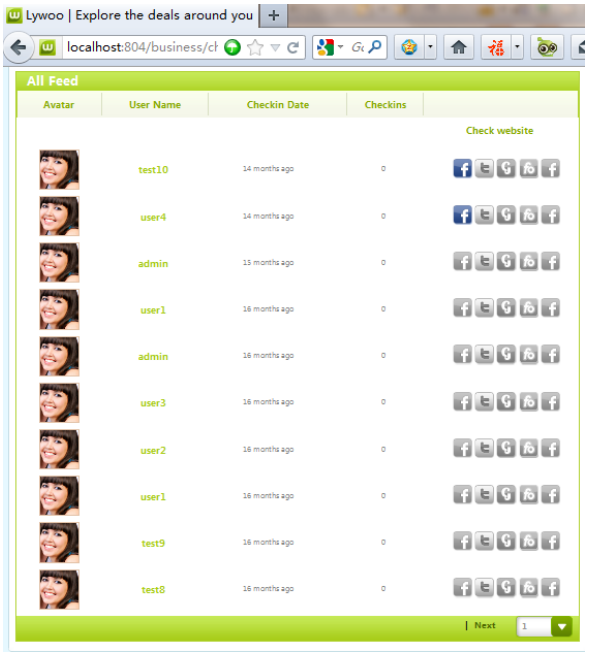

图 5-27 商家后台签到统计页面

2、商家后台优惠券统计页面

商家后台优惠券统计运行页面，如图 5-28 所示：

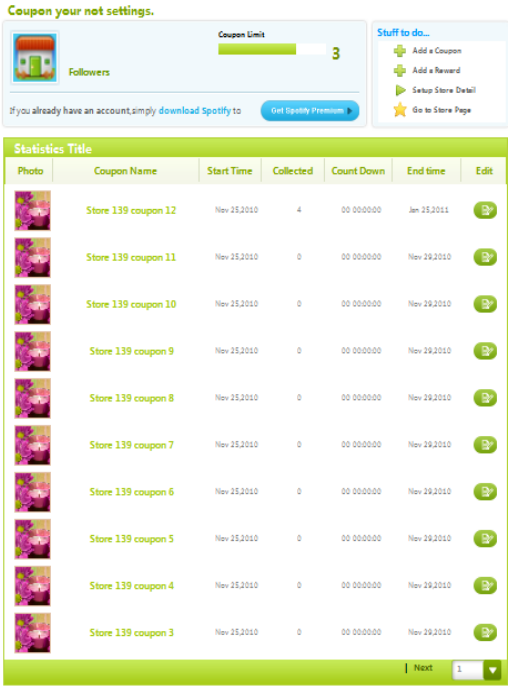

图 5-28 商家后台优惠券统计页面

3、商家后台活动统计页面，如下图 5-29 所示：

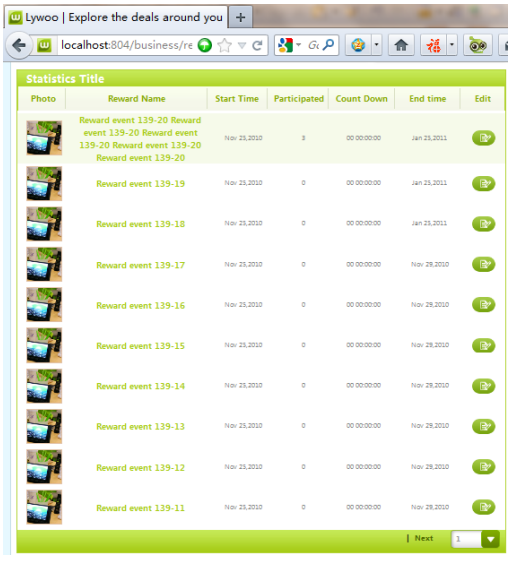

图 5-29 商家后台活动统计页面

4、商家后台活动统计页面，如下图 5-30 所示：

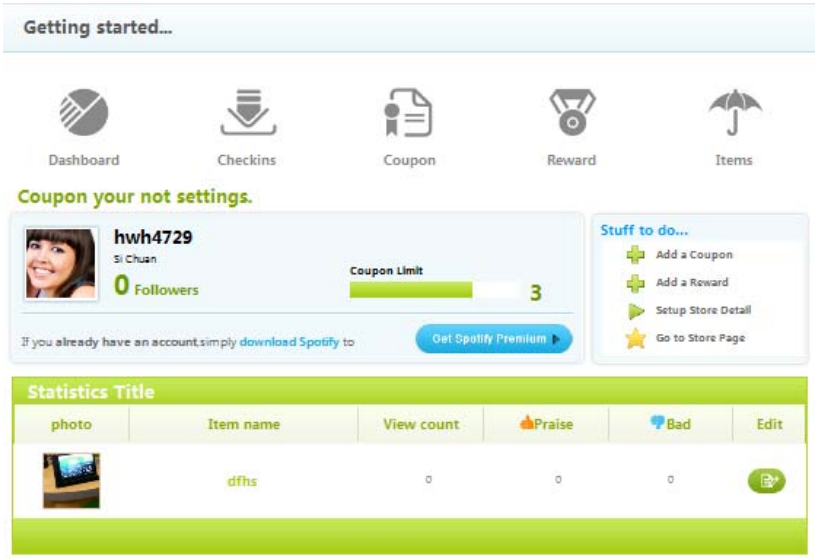

图 5-30 商家后台商品统计页面

5.6 本章小结

本章阐述了 Shopylife 的一些特殊功能模块在编码过程中所运用的技术和方法，包括检测用户位置，推送技术实现，二维码生成等，以及系统主要功能的实现和界面效果。

第六章 系统优化与测试

6.1 系统优化

在我看来，系统优化主要包括三个方面，第一个方面是服务器的优化，另一方面是数据库的优化，最后才是前端优化以及代码的优化。代码优化提高的效率与服务器优化相比不是一个数量级别，况且可以做一些缓存来减少与服务器的请求。所以本节将重点介绍下服务器优化和数据库优化和前端优化。

6.1.1 服务器优化

由于 Shopylife 的服务器使用的是 Apache，那么我们不得不讨论下 Apache 解析 PHP 的方式。总的来讲，Apache 运行 PHP 的方式有三种：CGI、FastCGI、Mod_PHP。Apache 默认是用自带的 Mod_PHP 模块运行的，而一般情况下 FastCGI 的效率要高于 Mod_PHP，而 CGI 的效率是最低的。之所以 FastCGI 的速度快是因为它是常驻内存的，不需要每一个请求都启动和初始化，极大的提升了 PHP 站点的性能。因此，我们安装了 FastCGI 以达到效率最优。

除此之外，Apache 配置中还有一些需要注意的地方：一些不必要的模块可以移除，控制最大连接数（Maxclients）的设置，让操作系统支持 sendfile 可以使 Apache 发送静态内容很快且占用的 CPU 时间很少，选择一个多路处理模块（MPM），安装恰当的 Expires，Etag 和 Cache_Control，使用 mod_gzip/mod_deflate，关闭 HostnameLookups，使用持久连接但不要把连接时间设置太高，禁用.htaccess 等。

6.1.2 PHP 编译缓存

因为 PHP 是解释型的语言，所以每个 PHP 文件在运行的时候都需要解释生成中间代码（opcode）然后再执行，同一个 php 文件被不同的用户访问，或者同一个用户在不同时间访问同一个 php 文件，每次打开时都需要重新编译，这样反复编译，很费时间。因此 Shopylife 采用了 APC 工具来实现 PHP 编译缓存。我们同时做压力测试请求 500 次，来对比使用 APC 之前和使用 APC 之后的效率如下图所示：

```
Complete requests:      500
Failed requests:        0
Write errors:           0
Total transferred:      87500 bytes
HTML transferred:       12500 bytes
Requests per second:    728.44 [#/sec] (mean)
Time per request:       13.728 [ms] (mean)
Time per request:       1.373 [ms] (mean, across all concurrent requests)
Transfer rate:          123.83 [Kbytes/sec] received
```

图 6-1 使用 APC 缓存前压力测试结果

```
Complete requests:      500
Failed requests:        0
Write errors:           0
Total transferred:      87500 bytes
HTML transferred:       12500 bytes
Requests per second:    1144.69 [#/sec] (mean)
Time per request:       8.736 [ms] (mean)
Time per request:       0.874 [ms] (mean, across all concurrent requests)
Transfer rate:          194.60 [Kbytes/sec] received
```

图 6-2 使用 APC 缓存后压力测试结果

由图 6-1、图 6-2 可知，使用了 APC 缓存后的每次请求时间（Time per request）明显小于使用 APC 缓存前的请求时间。

6.1.3 使用 MemCache 做数据缓存

Shopylife 系统采用 MemCache PHP 扩展提供数据缓存。它将减少访问数据库的次数，从而节省了大量的查询时间。我们对一次数据库访问 600 次，然后对使用 Memcache 和没有使用 Memcache 两种情况做个比较，发现用 Memcache 的话，执行速度几乎快 5 到 6 倍，如下图 6-3 所示。

```
Memcache used time:3.36253595352 senconds!
NO_memcache used time: 17.3725950718 senconds!
```

图 6-3 使用 Memcache 作为数据缓存前后的测试结果

6.1.4 Web 缓存

Shopylife 系统使用 Web 缓存来保存输出内容的副本，例如 Html 页面、图片、静态文件等，下一次请求相同的资源的话，Web 缓存会直接使用副本，而不会再向服务器发送请求。这样做可以减少请求时间，也可以减少网络带宽消耗。实现 Web 缓存可以使用 Http 协议中与缓存相关的消息头，如 Expires，Cache-Control，Last-Modified，Etag，Date，If-Modified-Since，If-No-Match 等。客户端缓存生效的常见流程，如下图 6-4 所示。

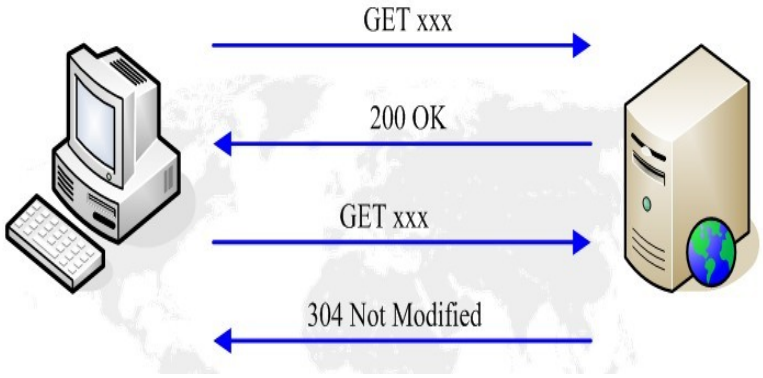

图 6-4 客户端缓存常见流程

代码如下：

```
<?php
//检查服务器文件上次更改的时间，默认为 0
$lastmodtime =
isset($_SERVER["HTTP_IF_MODIFIED_SINCE"])?intval(strtotime($_SERVER["HT
TP_IF_MODIFIED_SINCE"])):0;
$file = "./a.jpg";
$filetime = filemtime($file);
if($lastmodtime<$filetime)
{//如果文件本修改了
    $maxage = 8;
    //指定响应过期时间
    header("Expires:".gmdate('D,d M Y H:i:s',time()+$maxage).'GMT');
    //控制缓存内容
```

```
        header('Cache-Control:public');
        header('Last-Modified:'.gmdate('D,d M Y H:i:s',$filetime).'GMT');//设置最后一次修改时间
        $etag = "asdfasdf";
        header('Etag:'.md5($etag));//设置资源校验值
        echo 'ok';
}else{//文件没有被修改
        header('Last-Modified:'.gmdate('D,d M Y H:i:s',$filetime).'GMT');
        header('HTTP/1.1 304 Not Modified');
}
```

进行第二次请求后浏览器头信息显示，如下图 6-5 所示。

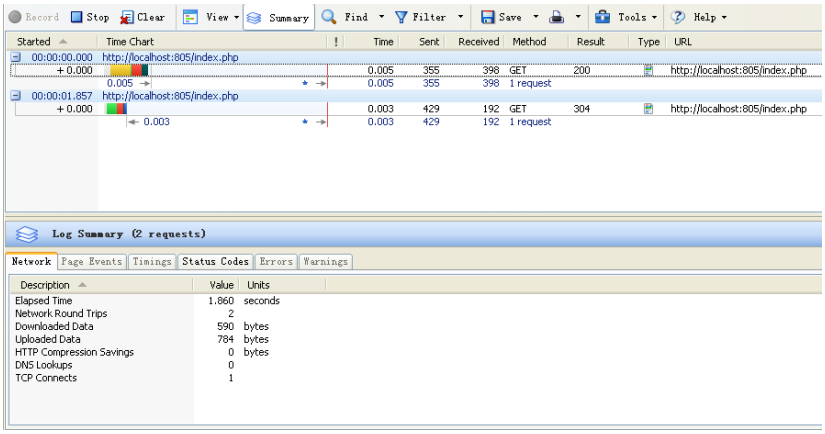

图 6-5 缓存后浏览器头信息

6.1.5 数据库优化

Shopylife 采用 MySQL 默认的存储引擎 MyISAM，每张 MyISAM 表被存放在三个文件(表格定义，数据文件，索引文件)中。另外，对于一些较频繁作为查询条件的字段创建了索引，从而极大地提高了以该字段作为检索条件的检索效率，降低检索过程中须要读取的数据量。同时尽可能避免复杂的 join 和子查询。

因为 MyISAM 相对简单，所以适合小型的读数据库多写数据库少的应用。在不同的应用中可能数据库配置各不相同，在 Shopylife 的数据库设置中，我做了如下优化：

（1）key_buffer_size：是用于索引块的缓存区大小，增加它可以更好的处理索

引，根据各表中索引的大小和目前的数据量以及负载情况来看，由于 MyISAM 表会使用操作系统的缓存来缓存数据，因此需设置“key_buffer_size”为可用内存的 35%，如果太大，系统将开始换页，这就会变得更慢了。

（2）sort_buffer：每个需要进行排序的线程分配一定大小的缓冲区。增加这值能加快 order by 或 group by 操作。默认数值是 2 兆，Shopylife 把它设置为 16M。

（3）query_cache：由于已经有了 memcache 作为数据缓存，所以我将“query_cache”设置很到 32Mb，因为设置太大会造成维护上的开销，这会导致 MySQL 变得更慢。

（4）thread_cache：设置它的目的在于通常的操作中无需创建新线程。因为每个线程的连接/断开都需要耗费时间，线程的创建和销毁的开销可能很大。由于 Shopylife 系统没有大量的跳跃并发连接，我通常至少设置为 16。

（5）table_cache：打开一个表的开销可能很大。MyISAM 每次打开都需要把 MYI 文件头标志该表正在使用中，这种操作也很耗时，所以通常需要加大缓存数量，使得足以很大限度地缓存打开的表。Shopylife 总共 32 张表，那么设置为 160（5*32）比较合适，因为每个线程都需要打开表，如果连接数比较大那么就加大它的值。

6.2 系统测试

一个 Web 应用程序基本完工后，需要经过测试才能上线。Web 应用程序测试与其他软件测试相比没有太大的区别，由于 Web 应用程序与用户直接相关，并承受大量的并发用户的访问，因此 Web 项目必须在达到预期的功能需求的情况下又同时具备可靠的性能。Web 测试主要包括：功能测试，性能测试，兼容性测试，可用性测试，客户端兼容性，安全性测试等。

6.2.1 功能测试

功能测试主要按照前面需求分析的要求测试是否完成了相应的功能。

1、链接测试：首先测试所有的页面 URL 是否按照规定链接到该链接的页面；然后，测试链接的页面是否存在；最后，保证整个系统中没有孤立的页面。测试结果如下图 6-6 所示，说明系统的所有链接都是正常的。

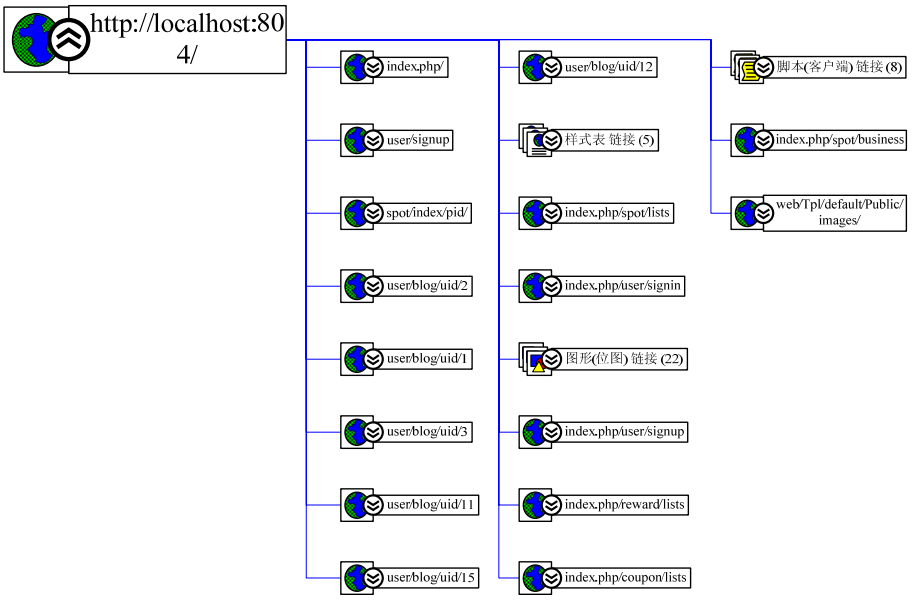

图 6-6 Shopylife 链接测试结果

2、表单测试：主要测试表单的显示是否完全，文本框超过规定文字是否允许提交，图片文件上传格式和大小错误是否能够提交成功，日期范围超过会不会提示错误，提交按钮是否允许同一时间点击多次等。通过使用 RoboForm 对系统注册、登录和优惠券添加三个表单的提交操作的完整性测试，检验提交给服务器的信息完全正确。

3、Cookie 测试：Shopylife 利用 Cookie 来存储用户的浏览历史记录，地理位置，是否自动登录等信息，因此必须测试所有 Cookie 是否能正常工作，测试用例描述为检查 Cookie 能否保持到预设的时间，刷新、关闭浏览器对 Cookie 有无影响，通过火狐的 Firecookies 插件查看这些 Cookie 信息是否已经加密，Cookie 保存的内容是否生效等。测试的结果是所有 Cookies 都能正常的工作。

4、数据库测试：数据库是否正确，表中字段是否完整，从测试服务器发布到主服务器之后，会不会因为环境不同而导致错误。通过检查数据库的错误日志，测试结果能够保证数据库的正常工作。

6.2.2 性能测试

1、连接速度测试

连接速度测试的主要内容是测试整个 Shopylife 系统的页面等待时间，如果等

待时间太长，用户可能会没有耐心而选择放弃。测试结果为所有 Shopylife 系统的页面第一次打开的平均时间为 3 秒钟。能够达到用户的要求。

2、压力测试

Web 服务器的性能测试主要测试的指标有：Requests per second（平均每秒钟响应次数），Complete Request（总的请求次数），Failed Request（失败的请求），Connection Times(连接的时间)等。我们可以通过使用 Apache 自带的压力测试工具“ab”来测试 Shopylife 系统的链接速度和赖受力。测试用例描述“ab -c 500 -n 500 http://localhost:804/”，表示有 500 个用户同时请求 500 次，测试结果如下图 6-7 所示，可得知平均每秒的响应次数为 20.86 次，总的请求次数为 500 次，失败的请求次数为 2 次,每个用户请求的响应时间大约为 24 秒,每次请求的响应时间约为 0.05 秒：

```
Server Software:        Apache/2.2.17
Server Hostname:        localhost
Server Port:            804

Document Path:          /
Document Length:        39153 bytes

Concurrency Level:      500
Time taken for tests:   23.971 seconds
Complete requests:      500
Failed requests:        2
   (Connect: 2, Receive: 0, Length: 0, Exceptions: 0)
Write errors:           0
Total transferred:      19769000 bytes
HTML transferred:       19576500 bytes
Requests per second:    20.86 [#/sec] (mean)
Time per request:       23971.371 [ms] (mean)
Time per request:       47.943 [ms] (mean, across all concurrent requests)
Transfer rate:          805.36 [Kbytes/sec] received

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:        0   26 119.8      0     997
Processing:  7526 12728 2488.8  14997   15037
Waiting:     7526 12678 2458.8  14906   14983
Total:       7526 12754 2465.0  14998   15037

Percentage of the requests served within a certain time (ms)
  50%  14998
  66%  15014
  75%  15019
  80%  15021
  90%  15028
  95%  15034
  98%  15035
  99%  15036
 100%  15037 (longest request)
```

图 6-7 Shopylife 压力测试结果

6.2.3 客户端兼容测试

1、平台测试

通过在用不同的操作系统上运行，主要包括 Windows2003、WinXP、Vista、Linux、Unix 等，看系统是否能兼容所有的这些系统。

2、浏览器测试

有许多种不同的浏览器，包括 IE，火狐，谷歌，Opera 等，不同的浏览器有不同的版本，因此还需要在每个浏览器下面打开 Shopylife 系统，看浏览器能否确保页面 CSS 的正常显示和 JavaScript 脚本是否兼容，并在窗口模式下拉伸或缩放看是否会造成影响。

除了以上两种外，还需要测试不同的客户端（包括宽屏显示器、普通显示器、笔记本显示器、手机 iPhone 等），以及同一客户端不同的分辨率下，测试结果均能保证系统的正常使用。

6.2.4 代码合法性测试

1、程序代码合法性测试

通过检查 Shopylife 系统中的 PHP 代码变量，常量，函数，接口，类方法的定义是否正确，控制符是否合法等。

2、显示代码合法性测试

显示代码主要包括 Html、CSS 等。通过 W3C 提供的 HTML、CSS 提供的验证标准工具（W3C validators，网址是：http://jigsaw.w3.org/css-validator/），系统的 HTML 和 CSS 均符合 W3C 标准。

6.2.5 安全性测试

测试是否只有管理员权限的用法才能进入 API 系统，只有商家账号才能进入商家管理系统，测试密码，账号正确和不正确的情况下能否登入系统。测试不同权限的用户是否只能操作该权限的操作。用户登录页面一段时间内不做任何操作，是否要求重新登录。进行 SQL 注入式的攻击（如 mm' or '2'>'1），测试是否有 XSS 漏洞，看是否对特殊字符进行转义或过滤。测试结果是整个系统没有任何安全漏洞。

6.2.6 接口测试

Shopylife 的 API 系统只是一个内部接口，只有系统管理员才有权限进行请求数据，提交信息等操作。在测试的时候，尝试在处理过程中中断事务时，系统能否正确处理。经过大量测试所有数据接口都能返回我们预期的结果。

6.3 本章小结

本章首先对系统进行了优化，通过 FastCGI 的方式来运行 PHP，利用 APC 缓存工具来实现 PHP 编译缓存，使用 MemCache 做数据缓存，并通过优化数据库配置文件和建索引的方式来提高数据库效率。然后对系统进行了功能测试、性能测试、客户端兼容测试、代码合法性测试、安全性测试、接口测试，其中性能测试采用的是 Apache 服务器自带的 AB 测试工具。

第七章 总结与展望

7.1 总结

通过对 Shopylife 系统的搭建与设计，我实现了这个 Push 型的 LBS 平台，移动互联网的兴起极大的方便了用户获取信息，Shopylife 结合了传统互联网和移动互联网的优势，将即时信息推送到用户身边，改变了用户获取信息的方式，由主动寻找变为被动接受，这也为商家一直以来寻求的精准营销提供了契机。同时也减少了用户的查找信息的时间和网络流量费用。

该系统主要有以下几个优点：1、把线上的优势和线下的资源进行了有效的整合，形成全方位的广播机制。2、卖家和卖家的及时互动。3、社会化购物分享。4、基于地理位置的搜索。5、信息覆盖广，包括商户，餐饮，服饰，卖场，百货，购物广场，休闲娱乐，旅游，居家，快消，美容美发，电子等各大类别的会员卡。

本系统分为 WEB 前台、商家管理后台和手机客户端三大功能模块，每个功能模块又分为多个小的模块。虽然整个系统的功能已经非常强大了，但是目前的系统还是存在不足，例如场所数据库不全，需要靠用户添加场所，但如何鼓励用户和商家来添加这些地点有必须得有强大的激励机制。

7.2 展望

Shopylife 项目虽然开发完成，但是 Web 系统还没有完全开发注册，一些功能可能还有需要改进的地方，例如：

（1）目前的优惠券信息推送只体现了即时性，而没有考虑到用户是否真的喜欢这些信息。例如在一段时间可能推送出同类的信息，可能用户对第二次推送的信息需求就没有那么的强烈。希望下一版本能够多考虑下是否能融合个性化信息和多样性信息。

（2）目前的手机客户端还只有 iPhone 的，这明显不能满足大众用户的需求，接下来我们还要开发 Android 版本的，还有 blackberry 版本。

致　谢

首先我要感谢我的导师张翼成教授，本文的研究和写作是在张翼成老师的精心指导和热心帮助下完成的。在三年的学习中张翼成老师渊博的知识、严谨的治学态度和勤奋的敬业精神一直影响着我，将使我终身受益。在本文完成之际，谨向张翼成老师致以最真挚的谢意和崇高的敬意！

同时还要感谢所有教过我的老师，他们认真的工作作风和严谨的治学精神使我受益匪浅。感谢本文涉及到的研究领域的学者们，他们的研究成果使得我可以借用他们的经验，明确研究的方向。

感谢我的同学们，在三年的学习生活中他们给予了我很多的帮助和支持。感谢他们给予我很多的帮助和支持。

我还要感谢我的父母和家人，是他们的辛劳使我能够顺利完成学业，是他们的不懈支持和鼓励我才能不断的进取。

最后，谨向在百忙之中抽出时间评审本文的各位专家表示最衷心的感谢！

参考文献

[1] 王乐鹏,李春丽,王颖. Foursquare 模式及在中国的发展对策探讨[J]. 科技信息. 2010(23)

王春．3G 时代位置服务技术研究．通信技术，2010，(05)

[2] 魏武挥．Foursquare:社会化点评．新经济，2010，(06)

[3] 吕文龙．中国的 Foursquare 探险队．互联网周刊，2010，(12)

[4] 杨菲菲. LBS 位置服务发展趋势探究[J]. 科技信息. 2011(24)

[5] 猛犸．LBS 寻找自己的足迹．21 世纪商业评论，2010，(07)

[6] 程士安,陈思. 基于地理位置服务(LBS)技术平台的传播规律——以“街旁”为例解读技术赋予信息分享的新权力[J]. 新闻大学. 2010(04)

[7] 阿呆. O2O:移动商务发展的未来之路[J]. 通讯世界. 2011(05)

[8] 豆瑞星. 移动互联网:O2O 的天堂[J]. 互联网周刊. 2011(19)

[9] 李元勤．地理位置服务或催生下一个谷歌：数字通信，2010，(03)

[10] 罗巍. 基于位置服务的移动电子商务平台构建[J]. 中国科技信息， 2010，(02)

[11] 朱洪军，吴金荣．位置服务系统研究与实现．计算机与现代化，2010，(04)

[12] 曹海兵等．Push 型 LBS 应用的实现技术研究．计算机应用研究，2006，(10)

[13] 王翔,李庆华,张锋军. 基于 LBS 的智能推送技术研究[J]. 通信技术. 2011(12)

[14] 王帅，刘厚全．基于主动任务模型的 LBS 系统架构研究．大众科技，2010，(06)、

[15] 蔡巍. “推送”(PUSH)技术简介[J]. 中国信息导报. 1999(03)

[16] 张丽宁. 基于 Push 技术的图书馆网络信息服务[J]. 科技情报开发与经济. 2006(23)

[17] 王培凤. Push 技术与图书馆信息推送服务[J]. 科技情报开发与经济. 2005(10)

[18] Kamil A. Grajski,Ellen Kirk. Towards a Mobile Multimedia Age – Location-Based Services: A Case Study[J] ,2003

[19] P M Adams,G W B Ashwell,R Baxter. Location-Based Services — An Overview of the Standards[J] ,2003

[20] Woo-Jin Choi,Sirin Tekinay. Location-Based Service Provisioning for Next Generation Wireless Networks[J] ,2003

[21] The location-based services renaissance:A new formula for suc-cess[EB/OL] .http://www.trueposition.c om/lrc/whitepapers.php. 2005, .

[22] WANG Xiaoxiao,YAN Jie. Analysis on Public Transportation Influences on Real Estate

Development in East Lansing[J]. Geo-Spatial Information Science. 2011(01)

[23] Autere S,Kalm J,Lehtinen K, et al..Technologies for adaptive mobile service development: use cases and technology survey[OL].. . (2001)

[24] Satoh,I.A testing framework for mobile computing software. IEEE Transactions on Software Engineering . 2003

[25] Adusei I K,,Kyamakya K,Erbas F.Location-based services:Ad-vances and challenges[C]. . 2004

[26] Jaehyun Park,Sung H.Han,Sungjin Kang,Youngseok Cho,Yong S.Park,Ungyeon Yang,Dongsik Cho. WEB-BASED EVALUATION OF THE AFFECTIVE SATISFACTION TOWARD MOBILE PHONE DESIGNS[A]. Proceedings of 17th World Congress on Ergonomics[C]. 2009

[27] GO Y.An implementation of moving objectdatabases for location- based services[D]. . Feb 2004

[28] Adusei I K,Kyamakya K,Erbas F. Location-based services:Ad-vances and challenges[C] .Canadian Conference onElectrical and Computer Engineering. Canada. 2004, :1-7 .

[29] Sebastian Herden,Arman Mkrtchyan,Claus Rautenstrauch,et al. Personal information guide a platform with location based servi-ces for mobile powered E-commerce[C] .Proceedings of the 14thInternational Workshop on Database and Expert Systems Ap-plications. Washington DC,USA: IEEE Computer Society, 2003, :895-900 .

[30] Daniel Ralph,Stephen Searby. Location and personalisation:Delivering online and mobility services[M] .London,UnitedKingdom: Institution of Electrical Engineers, 2004, .

电子科技大学
UNIVERSITY OF ELECTRONIC SCIENCE AND TECHNOLOGY OF CHINA

工程硕士学位论文

ENGINEERING MASTER DISSERTATION