**编号**

JNDX_M

**本科生毕业设计（论文）**

**题目：** 基于智能推荐的高校内推网系统

物联网工程 学院 计算机科学与技术 专业

学 号 0304110107

学生姓名 高 帅

指导教师 蒋 敏 教授

讲师

二〇一五年六月

# 摘要

1. 本课题主要研究一个基于智能推荐的高校内推网系统。旨在以优秀的推荐算法解决公司和应届毕业生之间的双向需求问题。

以学长或者学姐在本系统上发布招聘信息，学弟或者学妹投递简历，学长挑选最为满意的简历内推到公司指定部门从而通知面试为主线，并着重于以组合推荐算法，主动为学长智能推荐较为符合招聘要求的学弟，并为投递的简历进行匹配度的客观公正且符合逻辑的评分，另外也为学弟推荐较为符合的职位，以节省查找职位的宝贵时间。更加简单的方式提升应聘双方的效率，从而更好的使本系统成为连接应届毕业生与企业的桥梁。

目前的社会招聘方式，应聘人是直接面向需求部门的，针对性较强，应聘成功率较高，整体发展趋势趋于平稳上升期，未来的招聘模式必定以针对性招聘为主。而校园招聘对于应届生方面的内部推荐较为缺少，本系统针对于应届生的内部推荐，以智能推荐作为内推系统必不可少的工具，最直接的使用到了组合推荐，组合起基于内容的推荐，基于关联规则的推荐，基于习惯的推荐以及基于知识的推荐等多数推荐方法,为应聘双方提供了直观的推荐百分值。

关键词： 高校；内推；智能；组合推荐；

# Abstract

This topic mainly studies the communication of Universities internal referral system based on intelligent recommendation. Excellent recommendation algorithm is designed to solve the problem of two-way between the needs of companies and graduates.

Sister school or seniors publish recruitment information on this system, school brother send their resumes, seniors choose to push the most satisfactory resume to designated departments and notice interview. This is the main line of the system. And focus on a combination of recommendation algorithm, Initiative to meet the recruitment requirements recommended school brother for seniors, and to resume delivery of objective and fair and logical matching score, there is also more in line with the positions recommended for school brother, in order to save valuable time, a more simple way to enhance the efficiency of both candidates, so that the system becomes connect graduates and enterprises bridge.

Current social recruitment, Candidates who are directly facing the demand sector, highly targeted, high success rate of candidates, the overall trend has stabilized on the rise, the future recruitment model will certainly base on targeted recruitment. While the campus recruiting for graduating students is more a lack of respect for internal referral, This system is recommended for final year students in the interior, Intelligent recommendation as essential tools in the push system, The most direct use to the combination of recommendation, combination from content-based recommendation, recommended based on association rules, based on the recommendation of the habit and recommended knowledge-based approach etc… provides an intuitive recommendation percentile for both sides.

Keyword: University; Recommendation; Intelligent; Recommended combination

目录

[**Abstract** ii](#_Toc418712665)

[第一章 绪论 1](#_Toc418712666)

[1.1引言 1](#_Toc418712667)

[1.2 一个实用的高校内推系统 1](#_Toc418712668)

[1.3 实现本系统所需的资源 2](#_Toc418712669)

[第二章 高校内推系统概述 5](#_Toc418712670)

[2.1 内推系统的发展 5](#_Toc418712671)

[2.1.1 国内内推系统的诞生 5](#_Toc418712672)

[2.1.2 国际内推系统的发展情况 5](#_Toc418712673)

[2.1.3 中国内推系统的发展情况 6](#_Toc418712674)

[2.2本课题的设计目标 6](#_Toc418712675)

[第三章 基于智能推荐的高校内推系统程序设计 7](#_Toc418712676)

[3.1 bootstrap概述 7](#_Toc418712677)

[3.1.1 什么是bootstrap 7](#_Toc418712678)

[3.1.2 booststrp在本系统中的应用 7](#_Toc418712679)

[3.2 Webx3.0的总体概述 7](#_Toc418712680)

[3.2.1 什么是webx 7](#_Toc418712681)

[3.2.2 webx的设计理念 8](#_Toc418712682)

[3.2.3 webx的层次 10](#_Toc418712683)

[3.2.4 本系统为什么要用Webx 11](#_Toc418712684)

[3.2.5 webx的优势 11](#_Toc418712685)

[3.3服务层框架Spring 11](#_Toc418712686)

[3.3.1 Spring的框架特征 11](#_Toc418712687)

[3.3.2 Spring的七个模块 12](#_Toc418712688)

[3.3.3 Spring于本系统的重要性 14](#_Toc418712689)

[3.4 数据层的Ibatis 15](#_Toc418712690)

[3.4.1 Ibatis的设计理念 15](#_Toc418712691)

[3.4.2 本系统为什么要使用Ibatis 18](#_Toc418712692)

[第四章 智能组合推荐模块的研究 19](#_Toc418712693)

[4.1 智能推荐的简介 19](#_Toc418712694)

[4.2 智能推荐系统的发展历程 20](#_Toc418712695)

[4.3 智能推荐系统的基本框架 21](#_Toc418712696)

[4.3.1基于内容的推荐 21](#_Toc418712697)

[4.3.2基于协同过滤的推荐 22](#_Toc418712698)

[4.3.3基于组合的推荐 23](#_Toc418712699)

[4.4 智能推荐系统的作用 23](#_Toc418712700)

[第五章 基于智能推荐的高校内推系统的开发 25](#_Toc418712701)

[5.1 界面设计 25](#_Toc418712702)

[5.1.1 首页设计 25](#_Toc418712703)

[5.1.2 个人信息界面设计 26](#_Toc418712704)

[5.2 实现用户注册 27](#_Toc418712705)

[5.3 实现用户登录 28](#_Toc418712706)

[5.5 实现发布内推 33](#_Toc418712707)

[5.6 实现简历编辑 34](#_Toc418712708)

[5.7 实现智能推荐 35](#_Toc418712709)

[5.8 实现接收简历 36](#_Toc418712710)

[第六章 小结 37](#_Toc418712711)

[6.1 系统功能概述 37](#_Toc418712712)

[6.2 系统存在的不足 37](#_Toc418712713)

[6.3 有待改进之处 37](#_Toc418712714)

[参考文献 38](#_Toc418712715)

[致谢 39](#_Toc418712716)

# 绪论

## 引言

基于智能推荐的高校内推系统是针对应届毕业生的找工作内推网站。主要研究如何通过本系统如何使企业招到好人才，好人才找到好企业。

目前的社会调研中，针对于应届生的企业直接与应届生交互的内推网呈现出一种巨大的需求。根据中国教育在线的统计，2014年全国高校毕业生规模创历史新高，达到727万人，其中江苏省高校毕业生近54.5万人，2014年成为“更难就业季”，而2015年高校毕业人数同比2014年又增加22万人，对应届毕业生提出了更大的就业挑战，而目前全国高校毕业生的就业率在70%上下波动，也就是有近225万人每年未能找到工作或面临失业。其中互联网行业占的比重较重，而在未来的几年，中国的高校毕业生人数，将以每年25万人左右的速度增长，就业压力也会随之加大，各大应届生求职网上的职位信息铺天盖地，应届生们往往对于简历都是目的浅的海投状态，目前社招的内推招聘形式，让求职人直接和公司联系起来，优化了很多环节，入职率更高，而应届生的内推在公司也是备受推荐的，各大好公司为争夺人才不断推出新政策，学长内推学弟入职可拿丰厚的奖金，从企业和求职者之间行车一个良性的需求环，由此显示出未来企业和应届生将普遍使用内推网的趋势。

着眼于企业和应届生之间的良好交互，将是未来高校的内推网站是企业和应届生的核心依赖所在。因此，研究实现一个较为实用的高校内推系统就显得尤其的有意义。

## 一个实用的高校内推系统

根据本课题的研究需要，我们以阿里巴巴这家公司的实习生内推流程为例，阿里巴巴公司每年的2到3月份是实习生招聘的内推时间，在3月中旬内推结束，进入后续环节，内推流程如下图（1.2.1），而传统的求职招聘流程还是以求职者为主动因素，在招聘网站或者宣讲会上确定钟意职位，投递简历，等待通知。主动求职的同时却是被动获得信息，信息不能够对称，导致求职效率较低。具体流程图如下图（1.2.2）：



图（1.2.1） 图（1.2.2）

由于各大公司为了抢夺人才资源，实习生和应届生内推都会进行的比较早，而往往公司里已经就职的学长和大多数高校的毕业生学弟们并没有直接的熟悉度，也并不知道学弟的技术能力如何，从而使双方都缺失内推机会，本系统专注于已入职学长和学弟之间的交互问题，以学校为基本点推广，增加同校学长和学弟的交互程度，从而达到双方内推成功的目的。

## 实现本系统所需的资源

本系统的实现主要借助于JetBrains公司所提供的集成编译器IDEA。IDEA 全称IntelliJ IDEA，是java语言开发的集成环境，IntelliJ在业界被公认为最好的java开发工具之一，尤其在智能代码助手、代码自动提示、重构、J2EE支持、Ant、JUnit、CVS整合、代码审查、 创新的GUI设计等方面的功能可以说是超常的。IDEA是JetBrains公司的产品，这家公司总部位于捷克共和国的首都布拉格，开发人员以严谨著称的东欧程序员为主。目前IDEA也是深受国内开发者喜爱的编译器之一，用户群也大大超过了Eclipse，它在代码编写的智能令人惊叹。

下图为IDEA14.0.3的集成编译环境和智能提示。



图1-2 Idea集成编译环境

下图为IDEA所提供的调试界面。

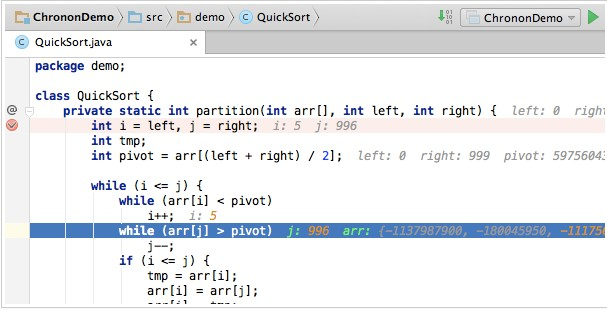


图1-3 Idea的调试环境

Idea的调试会在单行语句的后面显示出字段值，调试更加直观。本课题的开发还借助了Chrome的F12来debug调试和Fiddler4来抓包测试。

# 高校内推系统概述

## 内推系统的发展

### 国内内推系统的诞生

目前国内的大型内推网有两家：

一家是内推网（www.neitui.me）2013年4月28日上线第一个版本，这家网站创始人是黄小亮，曾在阿里巴巴和盛大网络工作，而内推网的灵感也是来自在阿里巴巴工作的时候感受到了阿里巴巴内部推荐的强大，也认识到了传统招聘渠道的一些问题，所以决定做一个纯中介的招聘平台，理想的方式下是由团队主管或者架构师直接和应聘者对接，创办内推网起初只有他们两个人，一个人兼职做技术、产品、运营、测试等，而另外一个人全职做技术开发和维护等。

另外一家是拉勾招聘(<http://www.lagou.com>) 2013年7月20日上线，上线的最初是走的国际板的LinkedIn模式，希望能够做出一版“中国版的LinkedIn”但是上线三个月后效果并不理想，拉勾网最终决定回到招聘本身，做职位内推也就是在对接双方需求的基础上，提供更具长远价值的服务。

在互联网的大潮下，一切都是可能被颠覆的，内推网的诞生就在一步一步颠覆传统的招聘行业。

### 国际内推系统的发展情况

内推在美国已经是一种较为流行的求职方式，其中google、apple、facebook、amazon和twitter这些知名互联网公司不仅倾向于向内推的应聘者提供橄榄枝，而且在应聘者入职成功后还会给推荐者一笔丰厚的奖金。目前较为出色的职业社交网站有LinkedIn和Classdoor。LinkedIn创建于2002年，2003年5月5日正式上线，目前已经成为全球最大的职业社交网站，致力于向全球职场人士提供沟通平台，并协助他们事半功倍，发挥所长。LinkedIn会员人数在世界范围内已经超过3亿，Linked有一个很大的优点，就是推荐信，求职者可以邀请朋友，教授，或者是公司职员帮助写推荐信，这会极大的增加档案的真实性，使技能与工作经历更受认可。并且可以帮助求职者扩展人脉网络，更利于公司的内部推荐。

### 中国内推系统的发展情况

目前中国的互联网大潮愈演愈烈，传统招聘网站在不断的被颠覆，很多时候传统的招聘网站都是HR发布招聘信息，应聘者投递简历，HR再挑出认为不错的简历录入到待面试系统，在铺天盖地的海投中，自身的简历很难被直接重视，而内推网就是绕过了猎头和HR的服务理念，目前的发展状态下内推网截止到目前每天有9万+的人来这个网站寻找机会，有3万+的知名互联网名企入住了内推网。而拉勾网相对于内推网的发展速度来说不相上下，仅仅在2014年6月底拉勾网上传的简历数量达到了25万左右，而总的投递量也接近了300万份。

未来的招聘系统中将更加直接的通过内推系统使招聘部门和应聘者直接对接，略过hr这一步，招聘效率更快。而以后的招聘将会成为企业和应聘者之间的一种服务。

## 本课题的设计目标

本课题致力于设计一款针对高校应届生的内推网站，就目前而言，高校每年的应届生会进入到很多公司企业，但是进入到企业后和高校的联系就基本断联了，学长对于母校，应届生和公司来说都是一笔丰厚的资源，公司希望在职员工可以推荐其优秀的学弟来公司，而学弟又倾向于对于学长的推荐。母校方面更在注重应届生的优秀就业率。本系统就是要把学长和学弟通过这个系统联系起来。通过学长的发布职位或者学弟的求职简历来简历联系，从而使学长对学弟的能力进行熟悉，并直接内推到公司部门，省去了在传统招聘系统中海投简历和不断跑路参加宣讲会的时间与精力。

# 基于智能推荐的高校内推系统程序设计

## bootstrap概述

### 什么是bootstrap

Bootstrap来自Twitter，是目前最受欢迎的前端框架，底层是基于 HTML、CSS、Javascript 的，它是一个CSS/HTML框架，并且提供了优雅的HTML和CSS规范，是由动态CSS语言less编写的，在jQuery的基础上进行了更为个性化和人性化的完善，形成了一套自己独有的风格，并且兼容大部分的JQuery插件，它的出现是为所有开发者、所有场景提供了较为方便, 简洁灵活，更加快捷的开发模式。目前在很多网站运用颇为广泛，成为了成为Github上热门的开源项目，目前很多公司都包含NASA的MSNBC（微软全国广播公司）的breakoing news都使用了该项目。

### booststrp在本系统中的应用

本系统的基本样式使用的都是bootstrap，包括导航栏置顶：导航栏永远停留在页头部分并在页头固定，输入框：并非简单的输入框，输入框具有记忆功能，能够记住你上一次的输入信息，然后双击输入框就会弹出你上次输入的信息并进行选择，页面按钮可以自定义按钮的大小并设定按钮的颜色，按钮形状相对来说比较圆滑更加人性化。另外页面的布局是响应式的布局，也就是当页面处于不同的宽度或不同的长度的时候页面会呈现不一样的界面，这就是响应式布局。

## Webx3.0的总体概述

### 什么是webx

Webx是一套基于Java Servlet API的通用Web框架。它在Alibaba集团内部被广泛使用。2001年，阿里巴巴内部开始使用Java Servlet作为WEB服务器端的技术，以取代原先的Apache HTTPD server和mod\_perl的组合。

2002年，选择Jakarta Turbine作为WEB框架，并开始在此之上进行扩展。

2003年，经过大约一年的扩展，框架开始成熟。我们私下称这个经过改进的Turbine框架为Webx 1.0。

2004年，借着淘宝网的第一次改版，我们正式推出了Webx 2.0。由于Turbine开源项目发展过于缓慢，我们不得不放弃它。Webx 2.0是从零开始完全重写的，仅管它仍然延续了Turbine的使用风格。

2004年11月，Webx 2.0和Spring框架整合。

从那以后，Webx 2.0一直在进化，但没有作根本性的改动。

2010年，Webx 3.0发布。Webx 3.0抛弃了Webx 2.0中过时的、从Turbine中发展而来的Service框架，直接采用Spring作为其基础，并对Spring作了重大改进。Webx 3.0完全兼容Webx 2.0的代码，只需要修改配置文件就可完成升级。

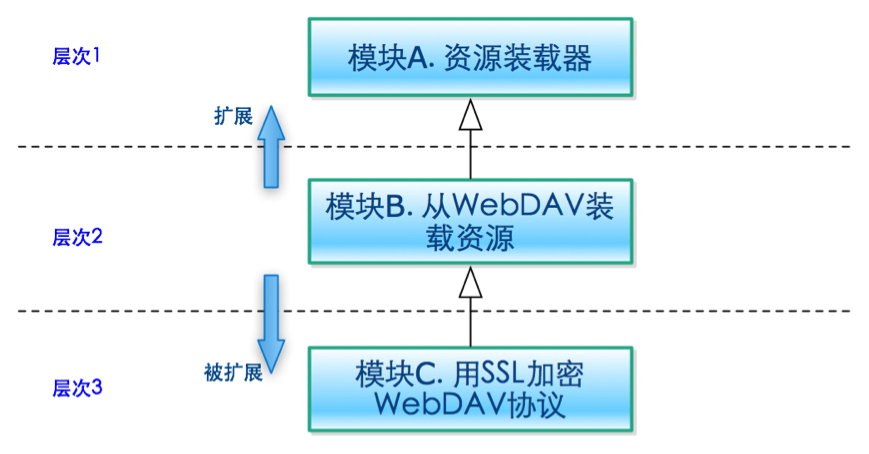
2010年底，Webx 3.0向社会开源。

### webx的设计理念

Webx的设计理念有两点，一点是框架化，一点是层次化。应用框架确定了整个应用的结构，而且框架允许你在不改变整体结构的接触上只有扩展功能，目前一些做的好的开源框架，都有一个共性 —— 它们并不是简单地实现Web应用所需要的功能（诸如Action、模板、表单验证等），而是把框架建立在另一个基础框架之上。这个基础框架的作用是：组装模块；提供扩展机制。建立在这种基础上的Web框架有很好的适应性和扩展性，可以应对Web应用不断变化和发展的需求。在早期的Turbine时是建立在Service基础上的，Webwork是建立在Xwork之上的，而如今的Spring MVC是建立在spring框架之上的。一个Web框架的好坏，往往不是由它所实现的具体功能的好坏决定的，而是由其所用的基础框架的好坏决定的。Webx建立在SpringExt的基础上 —— SpringExt是对Spring的扩展。Spring是当今主流的轻量级框架。SpringExt没有损失任何Spring的功能，但它能够提供比Spring自身更强大的扩展能力。

Webx的设计是层次化的，这也正好突出了模块的良好设计。

例如，模块B扩展了模块A，同时被模块C扩展。这样就形成了A、B、C三个层次。



如图所示，层次之间有如下的关系：

上层定义规则，下层定义细节；（上层、下层也可称为内层、外层）

上层是抽象的，下层是具体的；

越上层，越稳定（越少改变）；越下层，越易变。

依赖倒转（Dependency Inversion）。下层（具体）依赖上层（抽象），而不是上层依赖下层。

下层扩展上层时，不需要修改到上层的任何代码和配置。即符合开闭原则（Open-Closed Principle简称OCP – Open for extension, Closed for modification）。

每一层均可被替换。

层次化的设计，使软件中的每一个部分都可被增强或替换。

层次化不是自然而然的，而是需要精心的设计。设计一个层次化的组件，可以从下面几方面来考虑：

切分功能。每个组件专心做一件事。

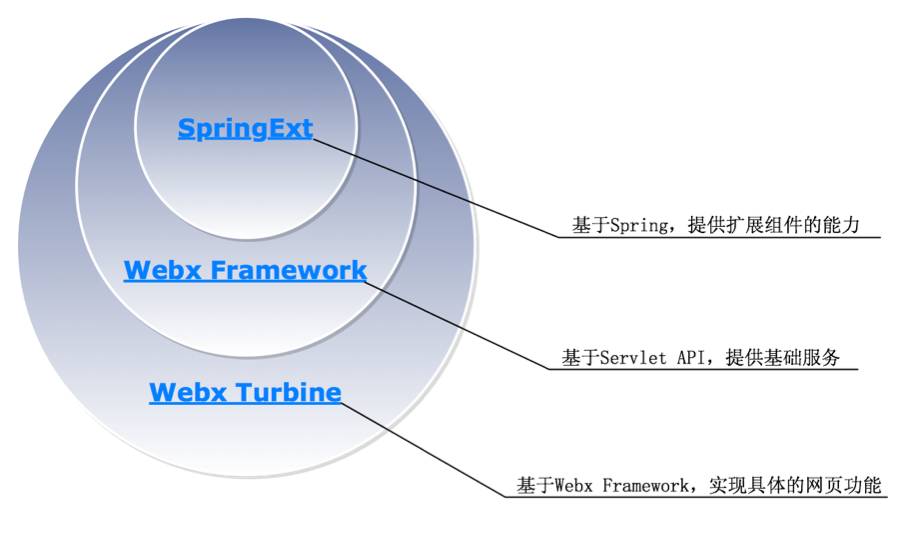
分析哪些会改变，哪些不会改变。不变部分固化在组件中，可能会改变的部分抽象成接口，以便扩展。

考虑默认值和默认扩展。默认值和默认扩展应该是最安全、最常用的选择。对于默认值和默认扩展，用户在使用时不需要额外的配置。

Webx鼓励层次化的模块设计，而SpringExt提供了创建和配置层次化组件的机制。

### webx的层次

很多用过Webx框架的人说起Webx，就想到：Webx如何处理页面、如何验证表单、如何渲染模板等等功能。事实上，这些只不过是Webx最外层、最易变、非本质的功能。Webx框架不仅鼓励层次化设计，它本身也是层次化的。你既可以使用全部的Webx框架，也可以只使用部分的Webx框架。大体上，Webx框架可以划分成三个大层次，如图所示。



**图 1.3. Webx的层次**

1.SpringExt：基于Spring，提供扩展组件的能力。它是整个框架的基础。

2.Webx Framework：基于Servlet API，提供基础的服务，例如：初始化Spring、初始化日志、接收请求、错误处理、开发模式等。Webx Framework只和servlet及spring相关 —— 它不关心Web框架中常见的一些服务，例如Action处理、表单处理、模板渲染等。因此，事实上，你可以用Webx Framework来创建多种风格的Web框架。

3.Webx Turbine：基于Webx Framework，实现具体的网页功能，例如：Action处理、表单处理、模板渲染等。

### ****本系统为什么要用Webx****

**从性能上讲Webx是一款很成熟的技术，经受住了天猫双十一的压力考验，并且在阿里巴巴多个系统的调用中具有很强的解耦性，对性能是一个很大的提升；**

**从结构上讲webx的框架很牢固，是一款层次化很强，很清晰的框架，它以“组件”为单位，易于后期系统的扩展，并且比没有扩展的spring更易使用，有因为层次化很清晰而足够开发者扩展。**

**从适合度上来讲，在阿里巴巴中，几乎每一个系统都在使用webx，webx在阿里巴巴的系统中是无处不在的，对于这款技术的上手速度也是较快的，而且在公司内部资源的基础下，后期会有更好的维护性。**

### ****webx的优势****

Webx是一种成熟可靠的技术，针对于阿里巴巴及属下网站而言，webx在阿里巴巴和淘宝网用了多年，目前也经受了多次双十一的高峰冲击，webx对于超大访问量的电子商务网站有很强的耐考验性。

另外webx在开放和扩展性方面也是很简介的，它对spring提供几乎无缝的支持，spring是当今主流的轻量级框架，webx3.0和springMVC一样，完全建立在spring框架之上，可以运用spring的所有特性，同时webx也对spring做了一些扩展，使spring bean不再是bean，而是升级为了“组件”，一个组件可以扩展另外一个组件，也可以被其他组件扩展，这种机制造就了webx的非常好的扩展性，且比未经扩展的spring更易使用，其次webx被设计成多个层次，层次之间分界线很清晰，每个层次都足够开发和已与扩展，开发者可以使用全部的webx也可以使用webx的任何一个层次。

## 服务层框架Spring

### ****Spring的框架特征****

Spring是一个开源框架，它由[Rod Johnson](http://baike.baidu.com/view/2192255.htm)创建。它是为了解决企业应用开发的复杂性而创建的。Spring使用基本的JavaBean来完成以前只可能由EJB完成的事情。然而，Spring的用途不仅限于服务器端的开发。从简单性、可测试性和松耦合的角度而言，任何Java应用都可以从Spring中受益。

Spring是一个轻量级的控制反转(IoC)和面向切面(AOP)的容器框架。

轻量——从大小与开销两方面而言Spring都是轻量的。完整的Spring框架可以在一个大小只有1MB多的JAR文件里发布。并且Spring所需的处理开销也是微不足道的。此外，Spring是非侵入式的：典型地，Spring应用中的对象不依赖于Spring的特定类。

控制反转——Spring通过一种称作控制反转（IoC）的技术促进了松耦合。当应用了IoC，一个对象依赖的其它对象会通过被动的方式传递进来，而不是这个对象自己创建或者查找依赖对象。你可以认为IoC与JNDI相反——不是对象从容器中查找依赖，而是容器在对象初始化时不等对象请求就主动将依赖传递给它。

面向切面——Spring提供了[面向切面编程](http://baike.baidu.com/view/1865230.htm)的丰富支持，允许通过分离应用的业务逻辑与系统级服务（例如审计（auditing）和事务（transaction）管理）进行内聚性的开发。应用对象只实现它们应该做的——完成业务逻辑——仅此而已。它们并不负责（甚至是意识）其它的系统级关注点，例如日志或事务支持。

容器——Spring包含并管理应用对象的配置和生命周期，在这个意义上它是一种容器，你可以配置你的每个bean如何被创建——基于一个可配置原型（prototype），你的bean可以创建一个单独的实例或者每次需要时都生成一个新的实例——以及它们是如何相互关联的。然而，Spring不应该被混同于传统的重量级的EJB容器，它们经常是庞大与笨重的，难以使用。

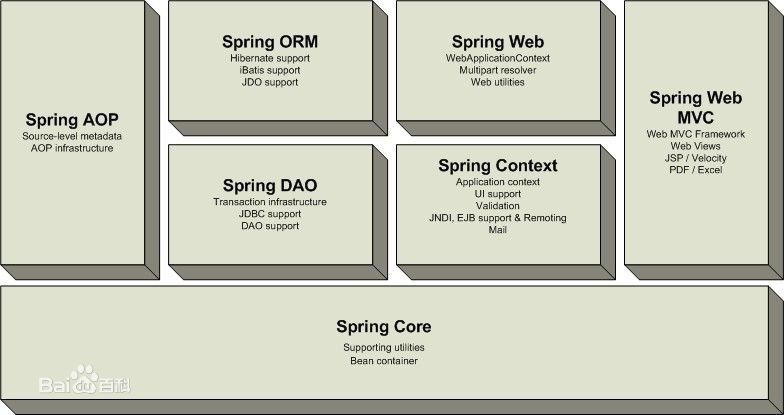
框架——Spring可以将简单的组件配置、组合成为复杂的应用。在Spring中，应用对象被声明式地组合，典型地是在一个XML文件里。Spring也提供了很多基础功能（事务管理、持久化框架集成等等），将应用逻辑的开发留给了你。

所有Spring的这些特征使你能够编写更干净、更可管理、并且更易于测试的代码。它们也为Spring中的各种模块提供了基础支持。

Spring适用于任何java应用,使用基本的javaBean代替EJB，并提供了更多的企业应用功能。目的是解决企业营业开发的复杂性。

### ****Spring的七个模块****

Spring框架由七个定义明确的模块组成（图1.1）。



（Spring框架概览图1.1）

如果作为一个整体，这些模块为你提供了开发企业应用所需的一切。但你不必将应用完全基于Spring框架。你可以自由地挑选适合你的应用的模块而忽略其余的模块。

就像你所看到的，所有的Spring模块都是在核心容器之上构建的。容器定义了Bean是如何创建、配置和管理的——更多的Spring细节。当你配置你的应用时，你会潜在地使用这些类。但是作为一名开发者，你最可能对影响容器所提供的服务的其它模块感兴趣。这些模块将会为你提供用于构建应用服务的框架，例如AOP和持久性。

**核心容器**

这是Spring框架最基础的部分，它提供了依赖注入（DependencyInjection）特征来实现容器对Bean的管理。这里最基本的概念是BeanFactory，它是任何Spring应用的核心。BeanFactory是工厂模式的一个实现，它使用IoC将应用配置和依赖说明从实际的应用代码中分离出来。

**应用上下文（Context）模块**

核心模块的BeanFactory使Spring成为一个容器，而上下文模块使它成为一个框架。这个模块扩展了BeanFactory的概念，增加了对国际化（I18N）消息、事件传播以及验证的支持。

另外，这个模块提供了许多企业服务，例如电子邮件、JNDI访问、EJB集成、远程以及时序调度（scheduling）服务。也包括了对模版框架例如Velocity和FreeMarker集成的支持。

**Spring的AOP模块**

Spring在它的AOP模块中提供了对面向切面编程的丰富支持。这个模块是在Spring应用中实现切面编程的基础。为了确保Spring与其它AOP框架的互用性，Spring的AOP支持基于AOP联盟定义的API。AOP联盟是一个开源项目，它的目标是通过定义一组共同的接口和组件来促进AOP的使用以及不同的AOP实现之间的互用性。通过访问他们的站点，你可以找到关于AOP联盟的更多内容。

Spring的AOP模块也将元数据编程引入了Spring。使用Spring的元数据支持，你可以为你的源代码增加注释，指示Spring在何处以及如何应用切面函数。

**JDBC抽象和DAO模块**

使用JDBC经常导致大量的重复代码，取得连接、创建语句、处理结果集，然后关闭连接。Spring的JDBC和DAO模块抽取了这些重复代码，因此你可以保持你的数据库访问代码干净简洁，并且可以防止因关闭数据库资源失败而引起的问题。

这个模块还在几种数据库服务器给出的错误消息之上建立了一个有意义的异常层。使你不用再试图破译神秘的私有的SQL错误消息！

另外，这个模块还使用了Spring的AOP模块为Spring应用中的对象提供了事务管理服务。

**对象/关系映射集成模块**

对那些更喜欢使用对象/关系映射工具而不是直接使用JDBC的人，Spring提供了ORM模块。Spring并不试图实现它自己的ORM解决方案，而是为几种流行的ORM框架提供了集成方案，包括Hibernate、JDO和iBATIS SQL映射。Spring的事务管理支持这些ORM框架中的每一个也包括JDBC。

**Spring的Web模块**

Web上下文模块建立于应用上下文模块之上，提供了一个适合于Web应用的上下文。另外，这个模块还提供了一些面向服务支持。例如：实现文件上传的multipart请求，它也提供了Spring和其它Web框架的集成，比如Struts、WebWork。

**Spring的MVC框架**

Spring为构建Web应用提供了一个功能全面的MVC框架。虽然Spring可以很容易地与其它MVC框架集成，例如Struts，但Spring的MVC框架使用IoC对控制逻辑和业务对象提供了完全的分离。

它也允许你声明性地将请求参数绑定到你的业务对象中，此外，Spring的MVC框架还可以利用Spring的任何其它服务，例如国际化信息与验证。

**Spring框架Web页面乱码问题**

在做java Web 项目时，乱码问题时常都会出现，解决方法也不尽相同，有简单也有复杂的；如果加入了Spring框架之后就不一样了，可以采用Spring框架自带的过滤器CharacterEncodingFilter，这样可以大大减轻了我们的工作量，即简单方便又容易理解，配置方式如下：在web.xml文件中filter的位置加上如下内容：

<filter>

<filter-name>encodingFilter</ filter-name >

<filter-class>

org.springframework.web.filter.CharacterEncodingFilter

</filter-class>

<init-param>

<param-name>encoding</param-name>

<param-value>UTF-8</param-value>

</init-param>

<init-param>

<param-name>forceEncoding</param-name>

<param-value>true</param-value>

</init-param>

</filter>

<filter-mapping>

<filter-name>encodingFilter</filter-name>

<url-pattern>\*</url-pattern>

</filter-mapping>

### ****Spring于本系统的重要性****

**本系统使用的webx是基于Spring的，最主要的用到的Spring3.0的注解功能和MVC结构。**

注解配置可以充分利用 Java 的反射机制获取类结构信息，这些信息可以有效减少配置的工作。如使用 JPA 注释配置 ORM 映射时，我们就不需要指定 PO 的属性名、类型等信息，如果关系表字段和 PO 属性名、类型都一致，甚至无需编写任务属性映射信息——因为这些信息都可以通过 Java 反射机制获取。另外注解和 Java 代码位于一个文件中，而 XML 配置采用独立的配置文件，大多数配置信息在程序开发完成后都不会调整，如果配置信息和 Java 代码放在一起，有助于增强程序的内聚性。而采用独立的 XML 配置文件，程序员在编写一个功能时，往往需要在程序文件和配置文件中不停切换，这种思维上的不连贯会降低开发效率。因此在本系统中选择使用注解配置，而注释配置有进一步流行的趋势。Spring 2.5 的一大增强就是引入了很多注释类，现在您已经可以使用注释配置完成大部分 XML 配置的功能。在这篇文章里，我们将向您讲述使用注释进行 Bean 定义和依赖注入的内容。

Spring 2.5 引入了 @Autowired ，@Resource注释，它可以对类成员变量、方法及构造函数进行标注，完成自动装配的工作。来看一下使用@Autowired 进行成员变量自动注入的代码：

@Autowired  
private UserManager userManager;

如果想要对这个对象进行注入，还需要在xml中配置一段扫描的配置，如：

*<!--扫描类包以应用注解定义bean-->*

<context:component-scan base-package="com.jiangnan.\*"></context:component-scan>

在xml配置了这个标签后，spring可以自动去扫描base-pack下面或者子包下面的java文件，如果扫描到有@Component @Controller@Service等这些注解的类，则把这些类注册为bean。

可以注册为bean的类在类头都会加上@Component @Controller@Service其中之一个字段，这三个字段分别代表不同的注册含义。

@Component  
public interface UserManager {}

**这样配置完后，在启动spring容器的时候，spring会自动扫描base-back下面或者子包下面的java文件。然后把对应的类注册为bean，当这个对象被调用的时候，就会自动向这些对象注入bean，实现方式特别简单灵活。**

## [****数据层的Ibatis****](#_Toc137881443)

### [****Ibatis的设计理念****](#_Toc137881444)

iBATIS是一个混合式的解决方案（hybrid solution），借鉴了多种操作关系数据库的方法的理念。

iBATIS是一种data mapper。也就是一个映射层，在对象和数据库间传递数据，并保持两者与映射层本身相独立。.

Martin很好地区分了数据映射（Data Mapping）和元数据映射（Metadata Mapping），后者正是O/RM工具的依据，这种工具将数据库的表和列映射到应用程序中的类和字段（field），也就是说它将数据库的元数据映射到类的元数据。图2.1显示了类和数据库表的O/R 映射的情形。在这种情况下，类的每个字段映射到了表中的一个相应的列。

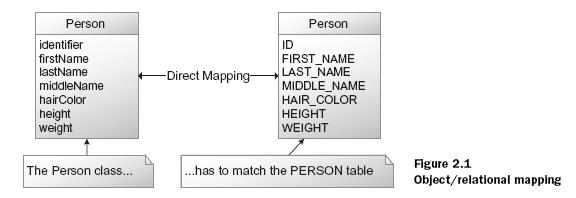


图2.1

iBATIS则与之不同，它不是直接在类与数据表或字段与列之间进行关联，而是把SQL语句的参数（parameter）和返回结果（result）映射至类。iBATIS是处于类和数据表之间的一个中间层，这使得它在类和数据表之间进行映射时更加灵活，而不需要数据库模型或对象模型（object model）的任何修改。这个中间层实际上就是SQL，它使得iBATIS能够更好地分离数据库和对象模型的设计，这样就相对减少了两者间的耦合。图2.2说明了iBATIS如何使用SQL映射数据。

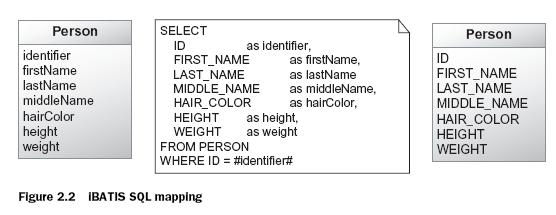


图2.2

从图2.2可以看到，iBATIS的映射层正是SQL。开发者只需要关心SQL，iBATIS会为您处理类的属性和数据表的列之间的映射。有鉴于此，同时也为了消除与其它各种映射方式引起的混淆，开发团队通常称这种Data Mapper为SQL Mapper。

SQL 映射是ibatis的一个重要设计理念，任何SQL语句都可看作是一组输入和输出。输入的值是参数（parameter），通常出现在WHERE子句中。输出的值则是出现在SELECT子句中的列。图2.3描述了这种观点。

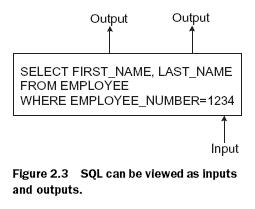
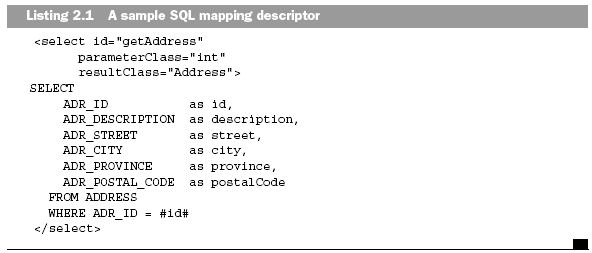


图2.3

这种方式的优势在于SQL语句给开发人员带来了很大的灵活性。我们可以轻松地操作数据使之与对象模型匹配而无需修改后台的数据库设计。此外，开发人员可以使用内置的数据库函数或存储过程来返回多个不同的表或结果，SQL的强大能力变得信手拈来。

iBATIS使用一个简单的XML描述文件来映射SQL语句的输入和输出。列表2.1显示了一个描述文件的示例。



列表2.1

此处可以看到一条SELECT SQL语句，它返回的是地址信息。根据<select>元素我们可以了解它接受一个整型数作为参数，也就是WHERE子句中标记为#id#的部分。我们也可了解结果为Address类型的一个实例，这里假定Address类包含了与SELECT子句中每一列的别名名称相同的属性。例如，别名为id的列将会映射至Address类的id属性。信不信由你，这就是映射一条接受整型参数、返回Address对象的SQL语句所需的全部了。执行这条语句使用的Java代码是：

Address address = (Address) sqlMap.queryForObject("getAddress", new Integer(5));

我们的SQL映射方式可移植性很强，可以应用到任何特性完整的编程语言。

### [****本系统为什么要使用Ibatis****](#_Toc137881445)

* **简单**

iBATIS是当前公认的最简单的持久层框架之一。简单是iBATIS团队设计目标的核心，其重要性几乎要超过其它任何方面。它的简单是通过它构建的基础来达到的,iBATIS对于数据库管理员和SQL开发人员来说也是简单的。几乎任何拥有SQL编程经验的开发人员都很容易理解iBATIS的配置文件。

* **生产力**

一个好框架要考虑的第一要旨是使开发人员更具生产力。通常框架会处理一些通用任务，减少重复性的编码，解决复杂的架构问题。在一项由意大利Java用户组进行的案例研究（http://www.jugsardegna.org/vqwiki/jsp/Wiki?IBatisCaseStudy）中，Fabrizio Gianneschi发现iBATIS可以减少持久层的代码达62％之多。SQL依然需要手工编写，但如您先前所见，SQL不是问题——不管您是使用JDBC还是ADO.NET,因为SQL是都可以优秀执行的。

* **性能**

事实上，在较低层次上来看，任何框架都会存在或多或少的性能损失。不过在当今应用程序开发中，更为重要的是，如何从数据库中获取数据，何时获取它，以及获取的频率。例如，从数据库中获取分页后的列表数据能显著地提升应用程序的性能，因为这样就避免了一次加载过多的数据。类似的，使用像延迟加载（lazy load）这样的特性可以避免在给定的用例下加载不必要的数据。另一方面，如果我们确定需要加载复杂的对象属性，而这些属性来自于多个数据表，那么使用单条SQL来加载数据也可以极大地改善性能。iBATIS提供了多种性能优化策略，通常要以简单的方式来配置iBATIS，但它的性能会像JDBC一样好，甚至更好。另一个需要考虑的地方是，并不是所有的JDBC代码都能编写得很好。JDBC是一个很直接面对数据库的API，要想正确使用需要注意很多地方，而且需要重复写很多代码。导致其性能甚至不如iBATIS。

* **分离关注点**

在典型的JDBC代码中，有时会看到数据库资源如连接、结果集散布在程序各个层中。在一些糟糕的程序中，我们会看到数据库连接、语句出现在表现层。这是最糟糕的设计。目前的程序开发，应用程序分层的是很重要的。我们看到了应用程序如何在较高层次上分层，持久层是如何处于中间层次的。iBATIS提供了此种分层的支持，它会管理所有数据持久相关的资源，如数据库连接，语句和结果集等。它提供了数据库无关的接口和API，帮助应用程序中的其它层能够与任何数据持久相关的资源保持独立。使用iBATIS，我们面对的是真正的对象，而不是任意的结果集。iBATIS使保持良好的分层变成一件容易的事。

* **分工**

有的数据库管理员很是珍爱他们的数据库，以至于不愿意其它人为其编写SQL。还有的人很擅长编写SQL，其他人都希望由他们来写SQL。不管处于何种情况，我们总是应该好好利用团队成员的优势。因为SQL语句和应用程序代码分离得非常清楚，SQL开发人员可以按其固有的方式进行开发，而不用担心什么字符串的拼接。即使有开发人员同时开发java代码和SQL，如果DBA想优化数据库的性能，只要说“让我看看SQL”。如果使用JDBC就没这么简单了，因为SQL往往包含在一连串的字符串拼接中，或者是由遍历和条件动态动态生成。使用O/RM会更糟糕，我们必须运行程序，然后在日志中输出语句，即使找到了，也不能做任何事情。iBATIS使得任何人都可以自由地开发、查看、修改SQL语句。

* **可移植性**

iBATIS可移植性是很强的。这得益于其相对简单的设计，它可以实现于几乎任何语言和平台上。在编写本书的时候，iBATIS支持三种最流行的开发平台：Java，Ruby和C#。

在当前配置文件还不是完全平台兼容的，但我们已有计划向这个目标靠近。更为重要的是，其*概念*和*方式*是可移植性很强的。这样我们所有应用程序的设计可以保持一致。对于语言和应用程序的类型来说，iBATIS比任何其它框架支持得都多。如果在您的程序中一致性非常重要，那么iBATIS将是很好的选择。

* **开源和可信度**

事实上，iBATIS是免费的，开源的软件。开源软件最大的优势之一是*可信*。开源代码可以直接查看源码的设计，以及对于后续的优化。还包含在遇到问题时如何快速的定位排查。

# 

# 智能组合推荐模块的研究

## 智能推荐的简介

互联网的出现和普及给用户带来了大量的信息，随着互联网在生活中的不断渗透，互联网信息对生活的狂风乱炸，各种信息数量和种类层不穷，虽然网络满足了用户在信息时代对信息的需求，但随着网络的迅速发展而带来的网上信息量的大幅增长，使得用户在面对大量信息时无法从中获得对自己真正有用的那部分信息，在浏览网页或者购买物品时会浪费大量的时间在一些不关紧要的页面上，从而不能精确定位到用户真正在意的问题，导致浪费大量的时候，又无法起到很好的反馈效果。对信息的使用效率反而降低了，这就是所谓的信息超载（informationoverload）问题。

解决信息超载问题一个非常有潜力的办法是推荐系统，它是根据用户的信息需求、兴趣等，将用户感兴趣的信息、产品等推荐给用户的个性化信息推荐系统。和搜索引擎相比推荐系统通过研究用户的兴趣偏好，进行个性化计算，由系统发现用户的兴趣点，从而引导用户发现自己的信息需求。一个好的推荐系统不仅能为用户提供个性化的服务，还能和用户之间建立密切关系，让用户对推荐产生依赖。智能推荐的特点就是根据用户的兴趣特点和行为，向用户推荐用户感兴趣的信息和商品，无论在任何场景下用户都会接触到智能推荐，智能推荐又叫个性化推荐，目前在电子商务方面用的比较多，比如淘宝而言，每个人进入淘宝首页看到的商品信息是不一样的，个性化推荐系统是建立在海量数据挖掘基础上的一种高级商务智能平台，以帮助电子商务网站为其顾客购物提供完全个性化的决策支持和信息服务。

推荐系统现已广泛应用于很多领域，其中最典型并具有良好的发展和应用前景的领域就是电子商务领域。同时学术界对推荐系统的研究热度一直很高，逐步形成了一门独立的学科。

## 智能推荐系统的发展历程

1995年3月，卡耐基.梅隆大学的RobertArmstrong等人在美国人工智能协会上提出了个性化导航系统Web Watcher;斯坦福大学的 MarkoBalabanovic等人在同一会议上推出了个性化推荐系统LIRA；

1995年8月，麻省理工学院的Henry Lieberman在国际人工智能联合大会（IJCAI）上提出了个性化导航智能体Letizia；

1996年， Yahoo 推出了个性化入口My Yahoo；

1997年，AT&T实验室提出了基于协作过滤的个性化推荐系统PHOAKS和Referral Web;

1999年，德国Dresden技术大学的Tanja Joerding实现了个性化电子商务原形系统TELLIM；

2000年，NEC研究院的Kurt等人为搜索引擎CiteSeer等加了个性化推荐功能；

2001年，纽约大学的Gediminas Adoavicius和Alexander Tuzhilin实现了个性化电子商务网站的用户建模系统1：1Pro；

2001年，IBM公司在其电子商务平台Websphere中增加了个性化功能，以便商家开发个性化电子商务网站。

2003年，Google开创了AdWards盈利模式，通过用户搜索的关键词来提供相关的广告。AdWords的点击率很高，是Google广告收入的主要来源。2007年3月开始，Google为AdWords添加了个性化元素。不仅仅关注单次搜索的关键词，而是对用户一段时间内的搜索历史进行记录和分析，据此了解用户的喜好和需求，更为精确地呈现相关的广告内容。

2007年，雅虎推出了SmartAds广告方案。雅虎掌握了海量的用户信息，如用户的性别、年龄、收入水平、地理位置以及生活方式等，再加上对用户搜索、浏览行为的记录，使得雅虎可以为用户呈现个性化的横幅广告。

2009年，Overstock（美国著名的网上零售商）开始运用ChoiceStream公司制作的个性化横幅广告方案，在一些高流量的网站上投放产品广告。 Overstock在运行这项个性化横幅广告的初期就取得了惊人的成果，公司称：“广告的点击率是以前的两倍，伴随而来的销售增长也高达20%至30%。”

2009年7月，国内首个推荐系统科研团队北京半分点信息科技有限公司成立，该团队专注于推荐引擎技术与解决方案，在其推荐引擎技术与数据平台上汇集了国内外百余家知名电子商务网站与资讯类网站，并通过这些B2C网站每天为数以千万计的消费者提供实时智能的商品推荐。

2011年9月，百度世界大会2011上，李彦宏将推荐引擎与云计算、搜索引擎并列为未来互联网重要战略规划以及发展方向。百度新首页将逐步实现个性化，智能地推荐出用户喜欢的网站和经常使用的APP。

## 智能推荐系统的基本框架

推荐系统有3个重要的模块：用户建模模块、推荐对象建模模块、推荐算法模块。通用的推荐系统模型流程如图。推荐系统把用户模型中兴趣需求信息和推荐对象[**模型**](http://baike.baidu.com/view/96500.htm)中的特征信息匹配，同时使用相应的推荐算法进行计算筛选，找到用户可能感兴趣的推荐对象，然后推荐给用户。本系统主要运用了以下三个方面的算法

### ****基于内容的推荐****

基于内容的推荐（Content-based Recommendation）是信息过滤技术的延续与发展，它是建立在项目的内容信息上作出推荐的，而不需要依据用户对项目的评价意见，更多地需要用机 器学习的方法从关于内容的特征描述的事例中得到用户的兴趣资料。在基于内容的推荐系统中，项目或对象是通过相关的特征的属性来定义，系统基于用户评价对象 的特征，学习用户的兴趣，考察用户资料与待预测项目的相匹配程度。用户的资料模型取决于所用学习方法，常用的有决策树、神经网络和基于向量的表示方法等。 基于内容的用户资料是需要有用户的历史数据，用户资料模型可能随着用户的偏好改变而发生变化。

基于内容推荐方法的优点是：

1）不需要其它用户的数据，没有冷开始问题和稀疏问题。

2）能为具有特殊兴趣爱好的用户进行推荐。

3）能推荐新的或不是很流行的项目，没有新项目问题。

4）通过列出推荐项目的内容特征，可以解释为什么推荐那些项目。

5）已有比较好的技术，如关于分类学习方面的技术已相当成熟。

缺点是要求内容能容易抽取成有意义的特征，要求特征内容有良好的结构性，并且用户的口味必须能够用内容特征形式来表达，不能显式地得到其它用户的判断情况。

### ****基于协同过滤的推荐****

协同过滤推荐 （Collaborative Filtering Recommendation）技术是推荐系统中应用最早和最为成功的技术之一。它一般采用最近邻技术，利用用户的历史喜好信息计算用户之间的距离，然后 利用目标用户的最近邻居用户对商品评价的加权评价值来预测目标用户对特定商品的喜好程度，系统从而根据这一喜好程度来对目标用户进行推荐。协同过滤最大优 点是对推荐对象没有特殊的要求，能处理非结构化的复杂对象，如音乐、电影。

[](http://baike.baidu.com/picture/2796958/2796958/0/a686c9177f3e6709118094b43ac79f3df9dc55fd.html?fr=lemma&ct=single)

协同过滤是基于这样的假设：为一用户找到他真正感兴趣的内容的好方法是首先找到与此用户有相似兴趣的其他用户，然后将他们感兴趣的内容推荐给此用 户。其基本思想非常易于理解，在日常生活中，我们往往会利用好朋友的推荐来进行一些选择。协同过滤正是把这一思想运用到电子商务推荐系统中来，基于其他用 户对某一内容的评价来向目标用户进行推荐。

基于协同过滤的推荐系统可以说是从用户的角度来进行相应推荐的，而且是自动的即用户获得的推荐是系统从购买模式或浏览行为等隐式获得的，不需要用户努力地找到适合自己兴趣的推荐信息，如填写一些调查表格等。

和基于内容的过滤方法相比，协同过滤具有如下的优点：  
　　1） 能够过滤难以进行机器自动内容分析的信息，如艺术品，音乐等。  
　　2） 共享其他人的经验，避免了内容分析的不完全和不精确，并且能够基于一些复杂的，难以表述的概念（如信息质量、个人品味）进行过滤。  
　　3） 有推荐新信息的能力。可以发现内容上完全不相似的信息，用户对推荐信息的内容事先是预料不到的。这也是协同过滤和基于内容的过滤一个较大的差别，基于内容的过滤推荐很多都是用户本来就熟悉的内容，而协同过滤可以发现用户潜在的但自己尚未发现的兴趣偏好。  
　　4） 能够有效的使用其他相似用户的反馈信息，较少用户的反馈量，加快个性化学习的速度。

虽然协同过滤作为一种典型的推荐技术有其相当的应用，但协同过滤仍有许多的问题需要解决。最典型的问题有稀疏问题（Sparsity）和可扩展问题（Scalability）。

### 基于组合的推荐

由于各种推荐方法都有优缺点，所以在实际中，组合推荐（Hybrid Recommendation）经常被采用。研究和应用最多的是内容推荐和协同过滤推荐的组合。最简单的做法就是分别用基于内容的方法和协同过滤推荐方法 去产生一个推荐预测结果，然后用某方法组合其结果。尽管从理论上有很多种推荐组合方法，但在某一具体问题中并不见得都有效，组合推荐一个最重要原则就是通 过组合后要能避免或弥补各自推荐技术的弱点。

在组合方式上，有研究人员提出了七种组合思路：  
　　1）加权（Weight）：加权多种推荐技术结果。  
　　2）变换（Switch）：根据问题背景和实际情况或要求决定变换采用不同的推荐技术。  
　　3）混合（Mixed）：同时采用多种推荐技术给出多种推荐结果为用户提供参考。  
　　4）特征组合（Feature combination）：组合来自不同推荐数据源的特征被另一种推荐算法所采用。  
　　5）层叠（Cascade）：先用一种推荐技术产生一种粗糙的推荐结果，第二种推荐技术在此推荐结果的基础上进一步作出更精确的推荐。  
　　6）特征扩充（Feature augmentation）：一种技术产生附加的特征信息嵌入到另一种推荐技术的特征输入中。  
　　7）元级别（Meta-level）：用一种推荐方法产生的模型作为另一种推荐方法的输入。

## 智能推荐系统的作用

个性化推荐的最大的优点在于，它能收集用户特征资料并根据用户特征，如兴趣偏好，为用户主动作出个性化的推荐。而且，系统给出的推荐是可以实时更新的，即当系统中的商品库或用户特征库发生改变时，给出的推荐序列会自动改变。这就大大提高了电子商务活动的简便性和有效性，同时也提高了企业的服务水平。

总体说来，一个成功的个性化推荐系统的作用主要表现在以下三个方面：

1） 将电子商务网站的浏览者转变为购买者：电子商务系统的访问者在浏览过程中经常并没有购买欲望，个性化推荐系统能够向用户推荐他们感兴趣的商品，从而促成购买过程。

2）提高电子商务网站的交叉销售能力：个性化推荐系统在用户购买过程中向用户提供其他有价值的商品推荐，用户能够从系统提供的推荐列表中购买自己确实需要但在购买过程中没有想到的商品，从而有效提高电子商务系统的交叉销售。

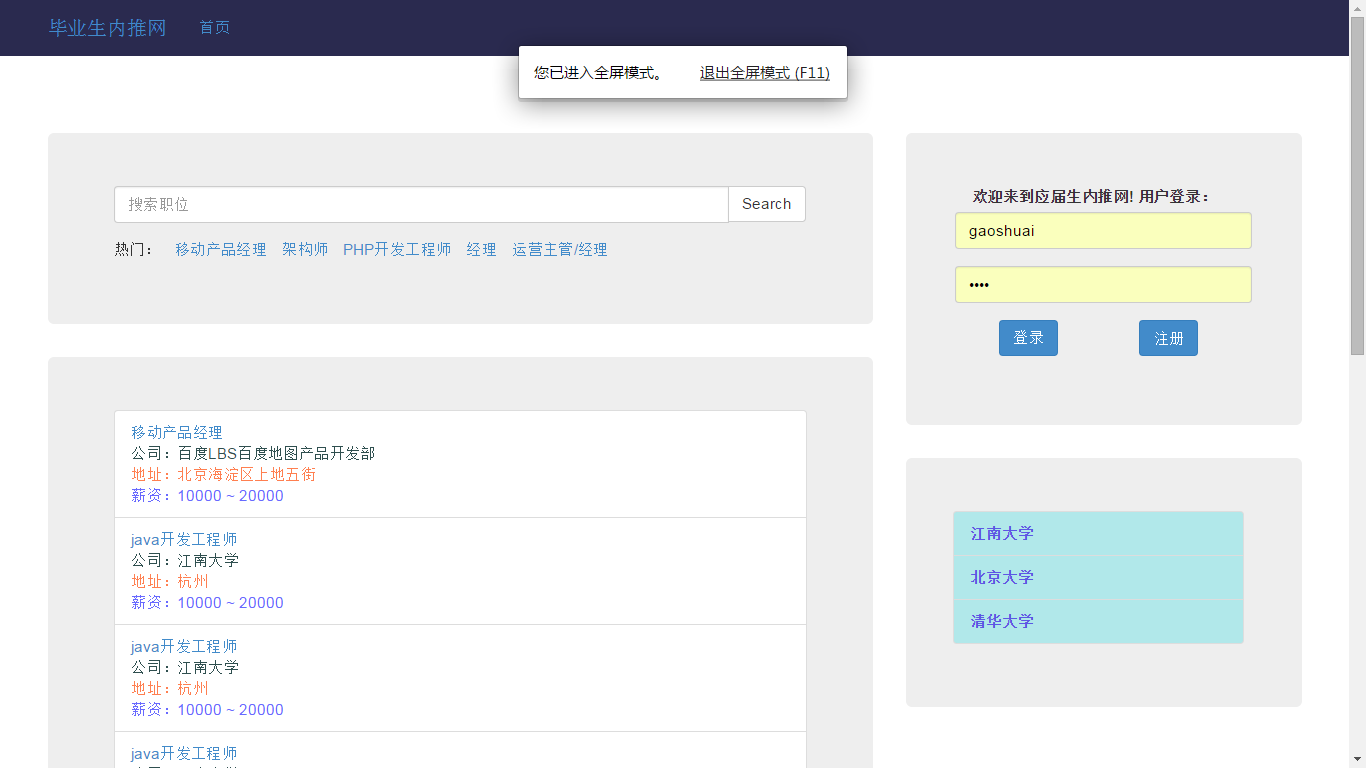
3）提高客户对电子商务网站的忠诚度：与传统的商务模式相比，电子商务系统使得用户拥有越来越多的选择，用户更换商家极其方便，只需要点击一两次鼠标就可以在不同的电子商务系统之间跳转。个性化推荐系统分析用户的购买习惯，根据用户需求向用户提供有价值的商品推荐。如果推荐系统的推荐质量很高，那么用户会对该推荐系统产生依赖。因此，个性化推荐系统不仅能够为用户提供个性化的推荐服务，而且能与用户建立长期稳定的关系，从而有效保留客户，提高客户的忠诚度，防止客户流失

个性化推荐系统具有良好的发展和应用前景。目前，几乎所有的大型电子商务系统，如Amazon、eBay等不同程度的使用了各种形式的推荐系统。国内方面，知名购物网站麦包包、凡客诚品、库巴网、红孩子等都率先选择了本土最先进的百分点推荐引擎系统构建个性化推荐服务系统。在日趋激烈的竞争环境下，个性化推荐系统能有效的保留客户，提高电子商务系统的服务能力。成功的推荐系统会带来巨大的效益。 另一方面，各种提供个性化服务的Web站点也需要推荐系统的大力支持，国内推荐系统领航者百分点科技就Web站点个性化内容推荐方面也做出了贡献，在信息爆棚的今天，实施个性化阅读势在必行。

# 基于智能推荐的高校内推系统的开发

## 界面设计

### 首页设计



该界面是用户未登录时的主界面，在webx中整个页面是一个大的layout。页面的最上方是固定的导航栏，右侧是登陆框，右下方是热门学校这些页面都是control。左上方是搜索框以及热门职位，左下方是一些职位信息，职位信息包含职位名称，公司名称，公司所在地和薪资范围，这两个页面是一个screen页面，用户通过左上方内嵌在首页layout的登录框登录。基于webx的机制，在本页面会可以记住用户所输入的密码。登录失败时会有弹窗提示，登录成功后会在当前页跳转，展示个人信息。参见5.3小节。

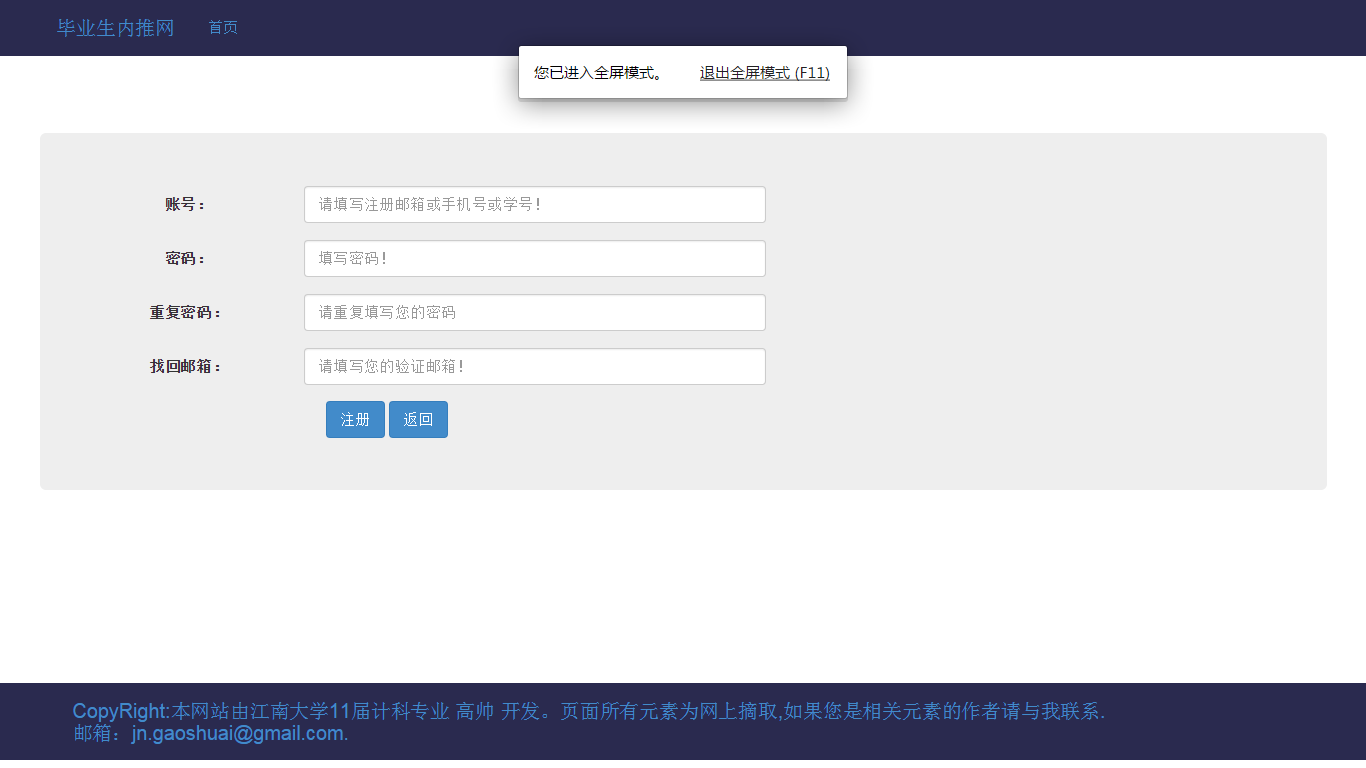
如果用户尚未注册，会提示该用户不存在，需要注册，点击注册按钮会跳转到注册，注册页面会在5.2小节进行介绍:

### 个人信息界面设计



进入个人信息页面时默认会展示个人信息，此页面左侧固定为几个栏目页面，分为个人信息，发布内推，简历编辑，智能推荐，投递记录，接受简历，发布记录这几个栏目，右侧为具体栏目对应的内容，个人信息页面如上图所示包含用户的真实姓名，联系邮箱，母校名称，自我介绍，联系QQ以及对于推荐条数的设置。填写完成后点击保存按钮，即可保存成功，如果保存失败会提示相关错误信息。

## 实现用户注册



用户注册，用户注册时填写注册账号，密码，并填写找回密码邮箱。实现过程如下：

public void doRegister(@Param("userId") String userId,

@Param("password") String password,

@Param("rePassword") String rePassword,

@Param("email") String email,

Context context

) {

if (!password.equals(rePassword)) {

context.put("errorMessage", "两次密码不相同，请重新输入！");

context.put("result", "failed");

return;

}

if (!checkEmail(email)) {

context.put("errorMessage", "邮箱格式不正确，请重新输入！");

context.put("result", "failed");

return;

}

User user = userManager.getUserByUserId(userId);

if (user != null) {

context.put("errorMessage", "该账号已存在，请换一个账号名注册哦！");

context.put("result", "failed");

} else {

Integer flag = userManager.register(userId, password, email);//数据库update的时候返回的是受影响的行数，如果行数大于0就表示成功了

if (flag > 0) {

context.put("result", "success");

} else {

context.put("errorMessage", "注册失败，请稍后再试！");

context.put("result", "failed");

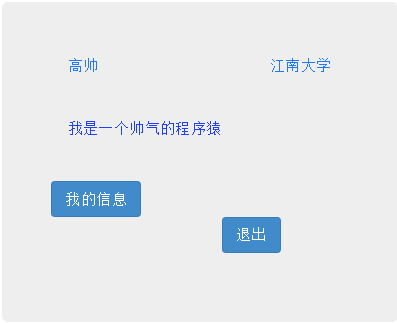
}

}

}

Webx根据@Param获得提交的字段数据，首先对两次输入的密码是否相同进行判断，将错误信息put到页面，然后返回，页面提示密码不相同，之后再检查输入的邮箱格式是否正确，如果不正确也同样会提示，第三点根据用户输入的账号查找数据库，如果能够获得到对象，则说明该账号是一家被注册过的，则返回，提示该账号一家被注册了，否则插入数据库，数据库使用的是ibatis的半自动化，插入数据库时该语句返回的是数据库受影响的行数，如果大于0说明插入成功了，如果不是，说明数据库可能暂时出现了问题，需要稍等一会再注册。注册成功后返回首页。

## 实现用户登录



用户填写账号和密码后登陆，如果输入的账户不存在，或者密码不对，则会有相应提示，用户登录成功后会在首页中原来登录的区域会展示为当前这个页面，当前页面会分散展示姓名，大学，和个人介绍。点击我的信息页面进入个人信息主页。点击退出，退出当前账号。具体实现代码为：

public void doLogin(@FormGroup("login")

@FormField(name = "loginError", group = "login")

@Param("userId") String userId,

@Param("password") String password,

Navigator navigator,

HttpSession session,

ParameterParser params,

Context context

) throws Exception {

User user = userManager.login(userId, password);

if (user != null) {

//在session中创建SessionUser对象

SessionUser sessionUser = (SessionUser) session.getAttribute(WebConstant.SESSION\_USER\_SESSION\_KEY);

if (sessionUser == null || sessionUser.hasLoggedIn()) {

sessionUser = new SessionUser(WebConstant.ACCESS\_REALM);

}

sessionUser.upgrade(userId);

session.setAttribute(WebConstant.SESSION\_USER\_SESSION\_KEY,sessionUser);

navigator.redirectTo(WebConstant.LOGIN\_RETURN\_DEFAULT\_LINK);

} else {

navigator.redirectTo(WebConstant.LOGIN\_RETURN\_DEFAULT\_LINK).withParameter("loginResult","failed").withParameter("errorMessage","用户名或密码不对！请重新登录!");

}

}

首先登录时先根据用户名和密码查询一下数据库，如果能够找到当前对象，说明该对象存在，如果找不到说明不存在。该对象存在时将该对象根据session\_key放入httpSession中，用户未退出本系统或未退出浏览器时Session中一直保存着该用户数据，直到该用户退出系统或者关闭浏览器，session中的数据将会被删除。

## 实现职位浏览与投递



根据webx的规则，分页查询职位信息，目前默认每页展示10个职位信息。

if (searchText == null) {

jobs = jobManager.getJobListWithPageSize(startRow);

sumResult = jobManager.getJobList().size();

} else {

jobs = jobManager.queryJobsBySearchText(searchText, startRow);

sumResult = jobs.size();

}

如果没有执行搜索，则直接分页查询职位信息。如果是搜索的则根据搜索关键字分页查询职位信息。



首页可点击职位信息进入职位详细信息页面，职位信息页面展示内退人，薪资，发帖时间，公司名称，公司地点以及内退人所在高校。在职位信息中展示职位的具体要求，在职位要求摘要中展示公司较为吸引人的一些摘要信息，比如公司具有健身房。点击发送简历按钮会将个人自己的个人简历发送给职位发布人。

public void doSendResume(@Param("id") Integer id,

HttpSession session,

Navigator navigator,

Context context

) throws Exception {

SessionUser sessionUser=

(SessionUser) session.getAttribute(WebConstant.SESSION\_USER\_SESSION\_KEY);

if(sessionUser==null){

context.put("result","failed");

context.put("errorMessage","你还没有登录，先到首页登录再投递该职位哦！");

navigator.redirectTo(WebConstant.LOGIN\_RETURN\_DEFAULT\_LINK);

return ;

}

DeliveryPost deliveryPost =

jobManager.getCheckSendInfoByJobIdAndUserId(id,sessionUser.getUserId());

System.out.println("deliveryPost="+deliveryPost);

if(deliveryPost!=null){

context.put("result","failed");

context.put("errorMessage","你已经投递过该职位啦，相同职位职能投递一次，赶快去看看其他职位吧。");

return ;

}

Integer flag= jobManager.addDeliveryRecord(sessionUser.getUserId(),id,DELIVER);//数据库update的时候返回的是受影响的行数，如果行数大于0就表示成功了

if(flag>0){

context.put("result", "success");

context.put("errorMessage", "投递成功啦，请静待学长联系你吧！");

}else{

context.put("result", "failed");

context.put("errorMessage", "投递失败，请稍后再试下哦，亲！");

}

}

如果投递职位时还没有登录会提示先登录，并自动跳转到登陆页，如果已经投递过该职位将不能重复投递相同的职位，满足以上要求后才可以正确发送简历。

## 实现发布内推



Job job = new Job();

job.setUserId(userId);  
job.setJobName(jobName);  
job.setCompany(company);  
job.setMinMoney(minMoney);  
job.setMaxMoney(maxMoney);  
job.setAddress(address);  
job.setJobReq(jobReq);  
job.setReqSummary(reqSummary);  
job.setUniversity(user.getUniversity());  
job.setEduBackground(eduBackground);  
job.setSpecialty(specialty);  
  
Integer jobId = jobManager.postJob(job);

if(jobId<0){  
 context.put("result", "failed");

return;

}  
Integer flag=jobManager.addDeliveryRecord(sessionUser.getUserId(),jobId,POST);

发布内推页面需要填写职位名称，公司，薪资，地点，职位要求和职位摘要以及学历要求。

且必填字段不能为空，每个字段会有验证，如果为空会做出验证并提示失败。验证通过后将职位信息记录到数据库。

## 实现简历编辑



Resume resume = new Resume();

resume.setUserId(userId);  
resume.setName(name);  
resume.setPhone(phone);  
resume.setEmail(email);  
resume.setBirthday(birthday);  
resume.setCity(city);  
resume.setMinMoney(minMoney);  
resume.setMaxMoney(maxMoney);  
resume.setExperience(experience);  
resume.setEduBackground(eduBackground);  
resume.setUniversity(university);  
resume.setSpecialty(specialty);  
Integer flag = userManager.editResume(resume);

简历编辑需要填写个人信息的若干字段，其中必填字段有姓名，手机，邮箱，出生年月，所在城市，经历和学历。这些字段会有相关的不为空验证，根据对象将个人信息set到对象的字段中，然后操作数据库，修改个人简历信息，个人简历保存时分为第一次创建和已有简历修改，当数据库中不存在个人简历时会直接创建一份个人简历信息，当数据库中已经存在当前登录者的个人简历时，会执行update个人简历而不是insert数据库。

## 实现智能推荐



public void execute(HttpSession session,

Context context) {

List jobs = new ArrayList();

SessionUser sessionUser =

(SessionUser)session.getAttribute(WebConstant.*SESSION\_USER\_SESSION\_KEY*);  
 List<Job> jobList = jobManager.getJobList();

Resume resumeInfo = userManager.getUserResumeByUserId(sessionUser.getUserId());

User user = userManager.getUserByUserId(sessionUser.getUserId());

List<DeliveryPost> jobNameList =

jobManager.getDeliveryNumsCountJobName(sessionUser.getUserId());  
 HashMap<String, Integer> hm = getJobNameNums(jobNameList);

List<AiResult<Job>> aiResultLists =

countAiNums(hm, jobList, resumeInfo).subList(0, user.getAiNums());

context.put("aiResultLists", aiResultLists);

}

推荐算法会根据用户的投递行为，是否和内推人在同一所大学，学历以及专业和职位要求来计算推荐指数。基于组合的推荐结合了用户的投递习惯，以及用户个人信息的内容，协同这两个方面对用户适合的职位进行推荐。

## 实现接收简历



public void execute(HttpSession session,

Context context) {

SessionUser sessionUser =

(SessionUser) session.getAttribute(WebConstant.SESSION\_USER\_SESSION\_KEY);

List<Job> jobs = jobManager.getJobInfoByUserId(sessionUser.getUserId());

List deliveryPosts = new ArrayList();

for (Job job : jobs) {

List deliveryPost = jobManager.getJobListByJobId(job.getId(), DELIVERY);

deliveryPosts.addAll(deliveryPost);

}

context.put("deliveryPosts", deliveryPosts);

}首先根据用户id查询到该用户发布的职位信息，然后根据职位信息在投递记录表中获取这些职位有哪些投递记录，进行展示。

# 小结

## 系统功能概述

本系统是一个基于智能推荐的内推网系统。TODO

## 系统存在的不足

由于自己的技术能力有限，因此这个系统存在诸多的不足或者说是有待完善的地方。以下根据当前主流的内推网系统列出我的这个系统的一些不足之处：

1. 目前界面的交互和视觉较差。
2. 用户个人简历只能是自主填写而不能够上传附件。

## 有待改进之处

1. 页面在视觉上仍有很大的提升余地，以及在交互上并不是很简洁。
2. 用户个人简历以附件word或者pdf文件存储。

# 参考文献

[1] （美）埃克尔著 陈昊鹏 译. java编程思想（Think in java）机械工业出版社，2007.

[2] 李刚.java疯狂讲义 电子工业出版社，2008.

[3] 陈雄华，林开雄. Spring3.0就这么简单 人民邮电出版社 2013.

[4] 沃尔斯，布雷登巴赫. Spring in Action 人民邮电出版社 2008

[5] （加）贝让 等著，叶俊 等译. IBATIS实战 人民邮电出版社 2008

# 致谢

首先要感谢我的导师蒋敏老师。蒋老师是一个特别特别认真的好老师，在课题设计的过程当中，蒋老师认真的为我们规划毕业设计时间点，帮我们耐心审查开题报告和检查论文翻译提出其中描述不清晰的点，指导我们认真按时完成毕业设计，为毕业设计的后期留下充足的时间，在我的毕设中给出很多专业的建议和指导。

其次要感谢我的现实习公司的师兄，公司中每天有很多的事情，有时间都忙到很晚很晚，毕设是在平时晚上的空余时间拼凑时间来完成的，界面不是很美观，功能也不是特别的完善，还需要做很多的修改，平时公司的事情很多，师兄知道我在做毕设，分配给我的任务就相对比较少些，所以可以在晚上空出大多数的时间来做毕业设计。

最后要感谢大学四年里我所遇到的好学长好老师们，从你们身上我学到了一个物联网专业弱校向外界展示的不屈与努力，也是因为你们，我才有耐力在实验室一行代码一行代码的书写我的大学。感谢大学四年里和我一起相处的所有同学们，我的室友，我是一个睡觉很容易受打扰的人，经过一段时间磨合，晚上大家都开始早睡，是你们陪我成长、陪我走过大学四年美丽的光阴。

真诚地感谢你们！