学 号 2012212117

密 级

哈尔滨工程大学学士学位论文

基于O2O的在线打印系统设计与实现

院（系）名　称：国家保密学院

专　业　名　称：信息安全（保密技术）

学　生　姓　名：刘彦辉

指　导　教　师：王红滨 讲 师

哈尔滨工程大学

2016年6月

**基于O2O的在线打印系统设计与实现**  **刘彦辉 哈尔滨工程大学**

学 号 2012212117

密 级

基于O2O的在线打印系统设计与实现

Design And Implementation Of Online Printing System Based On O2O

**学生姓名：**刘彦辉

**所在学院：**国家保密学院

**所在专业：**信息安全(保密管理）

**指导教师：**王红滨

**职称：**讲 师

**所在单位：**哈尔滨工程大学

**论文提交日期：**2016年5月

**论文答辩日期：**2016年6月

**学位授予单位：**哈尔滨工程大学

摘要

当前，有很多在线O2O网络平台通过计算机技术被开发出来。在日常生活中，人们可以接触到例如:在线O2O订餐平台、在线家政服务平台等。但并未出现在线O2O打印这个细分领域的可用平台。本课题以在线O2O打印这一细分领域为着眼点，设计了一种在线系统，用以连接打印社与普通用户。同时为了方便普通用户在线资源共享和相关信息交流，本课题为O2O平台配套建立了一个资源社区。另外，结合本平台自身优势，本课题也为本平台设计了一套平台内自由商品管理、发布、交易平台。

本文首先对O2O在线交易平台的发展和现状做了概括和介绍，然后分析了开发满足特定需求的O2O平台的必要性，并提出了本课题设计的出发点。之后，通过分析本课题的需求，划分出了各个功能模块。最后，对各个功能模块分别进行程序的开发与实现。

本平台在后台采用Sqlite建立存储用户资料及系统数据的数据库，并采用Django作为系统结构的主框架，同时使用zendstudio、Python IDLE作为后台编码环境。

关键词：O2O；在线打印；云存储；Django

**ABSTRACT**

Currently, many online O2O trading platforms are developed by computer science technology. In daily life, many O2O platforms are exposed to people, such as: online ordering O2O platform, online housekeeping O2O platform, etc. However, there is not any usable platform in the subdivided area of O2O printing platform. Focusing on O2O printing, the issue designs an online system to connect the print communities and common users. Also, in order to provide convenience for common users to share resources online and communicate relevant information, the issue establishes a resource community. In addition, combined with the advantages of the platform itself, the issue designs a free merchandise management, publishing, trading platform within the platform.

Firstly, the paper concludes the development and status of O2O online trading platform, and then analyzes the necessity to develop a O2O platform which meets the specific demands, and then proposes the starting point of the project design. After, different functional modules are divided according to the demand analysis of this project.

Finally, every functional module is developed implemented. In the platform, Sqlite is used to store users information and system data. Also, Django is developed as the main frame and zendstudio, Python and IDLE are used as coding environments.

**Key words:** O2O; Online printing; Cloud storage; Django

目 录

[第1章 绪论 1](#_Toc451862573)

[1.1 论文研究的背景和意义 1](#_Toc451862574)

[1.2 国内外研究现状 2](#_Toc451862575)

[1.3 论文研究的意义 3](#_Toc451862576)

[1.4 论文的组织结构 4](#_Toc451862577)

[第2章 系统开发环境及相关的技术 6](#_Toc451862578)

[2.1 开发环境概述 6](#_Toc451862579)

[2.1.1 PYTHON语言简介 6](#_Toc451862580)

[2.1.2 SQLITE 3.0简介 7](#_Toc451862581)

[2.1.3 PYTHARM.3、SUBLIME\_TEXT.3简介 8](#_Toc451862582)

[2.2 相关技术简介 8](#_Toc451862583)

[2.2.1 MVC与MTV模式 8](#_Toc451862584)

[2.2.2 DJANGO简介 11](#_Toc451862585)

[2.3 本章小结 13](#_Toc451862586)

[第3章 在线打印系统的分析设计 22](#_Toc451862587)

[3.1 系统需求分析 22](#_Toc451862588)

[3.2 功能模块的划分 24](#_Toc451862589)

[3.3 数据库设计 25](#_Toc451862590)

[3.3.1 数据库的设计 25](#_Toc451862591)

[3.4 模块设计 29](#_Toc451862592)

[3.4.1 登录模块的设计 29](#_Toc451862593)

[3.4.2 文件管理模块的设计 30](#_Toc451862594)

[3.4.3 商家后台模块的设计 32](#_Toc451862595)

[3.4.4 资源分享社区模块的设计 34](#_Toc451862596)

[3.5 本章小结 35](#_Toc451862597)

[第4章 在线打印系统的实现 36](#_Toc451862598)

[4.1 数据库的实现 36](#_Toc451862599)

[4.2 用户登录模块的实现 38](#_Toc451862600)

[4.3 文件管理模块的实现 39](#_Toc451862601)

[4.3.1 文件上传下载模块的实现 39](#_Toc451862602)

[4.3.2 文件夹创建和文件搜索模块的实现 40](#_Toc451862603)

[4.3.3 文件进入打印流程及查看详情模块的实现 41](#_Toc451862604)

[4.4 打印前设置模块的实现 42](#_Toc451862605)

[4.5 商家后台模块的实现 43](#_Toc451862606)

[4.5.1订单管理模块 43](#_Toc451862607)

[4.5.2店员管理模块 44](#_Toc451862608)

[4.5.3积分管理模块 44](#_Toc451862609)

[4.6商品编辑发布模块的实现 45](#_Toc451862610)

[4.7资源交流社区模块 45](#_Toc451862611)

[结论 47](#_Toc451862612)

[参考文献 48](#_Toc451862613)

[攻读学士学位期间发表的论文和取得的科研成果 49](#_Toc451862614)

[致谢 50](#_Toc451862615)

# 第1章 绪论

## 1.1 论文研究的背景和意义

1.1.1论文研究的背景

2012年是O2O元年，无论是成熟的传统企业、如火如荼的电子商务企业，还是以电信、银行、娱乐等为代表的与民生相关的企业，都在探索和践行O2O模式，因为O2O中孕育着极富创新性的商业模式[1]。O2O即Online To Offline,是指将线下的商务机会与互联网相结合，让互联网成为线下交易的展示平台。

首先介绍一下本课题研究的背景。计算机技术的发展时时刻刻地改变着人们的生活，而在线O2O交易平台就是其中的一个方面。许多优秀的O2O平台在人们的生活扮演者举足轻重的左右，它们无时无刻优化着人们的衣食住行。例如当前比较热门的O2O交易平台有嘀嘀打车、美团网、饿了吗等。O2O模式真正的将互联网连接一切资源和需求的属性发挥到一个较高的水平。而恰恰也正是这种属性将O2O模式的价值得以最大化。因此本课题选择基于O2O模式的在线交易系统设计，旨在挖掘被多数商业公司忽略掉的在线打印这一细分领域的剩余价值。当然也要以最好的计算机系统设计为目标不断优化当前在该方面有高频需求的用户的使用体验。

O2O在线交易平台的引入，可以提高商家和用户之间快速的线上信息匹配和线下的资源供给。在这个过程中，商家手中的打印设备、空间、耗材等资源得到了充分的展示和调度。从而将商家的接单率最大化、收益最大化。另外也将用户的购物流程最简化，时间消耗最小化。从而提高整个交易环节中各角色的社会必要时间效率。例如图1.1就是当前O2O模式最准确的描述。

图1.1 O2O原理模式图

在传统的商业模式中，无论客户直接到商家店铺直接进行商品浏览，消费还是互联网发展早期的在线购物商城都是本质上都是，作为商家提供一个固定的场所进行商品展示，消费者通过浏览发现符合自己需求的商品后直接下单支付，或现场同时完成商品浏览和支付，或同时在线完成商品浏览和支付。但是O2O最典型的一点就是用户可以在线进行商品浏览筛选，线下进行体验和消费[1]。这就是此系统开发的背景。

## 1.2 国内外研究现状

1、O2O 1.0

只是简单的将线上线下进行对接，充分利用互联网的信息快速传播特点把有相应需求的用户进行大规模快速集中，然后再快速的将商家信息进行整合传播，达到快速市场推广的作用。该阶段以普通用户单项自发的信息共享为主。但是作为整个交易流程中必备的平台方却很少有机会与用户直接接触，进而直接发生交易行为，或者说这种交易并未有传统的类似商场的集中交易场所，因此这个时期整个O2O市场以用户驱动为主导，驱动方面相对单一，发展速度也不尽人意。

2、O2O 2.0

最主要的升级就是向着服务性电商模式发展，有了相对完善的商品，服务，支付等流程，通过将电商模式，复制到现实的生活场景中来。传统的线下商铺的交易模式需要用户亲自上门体验、挑选、下单。这种传统的方式导致了传统的商铺交易必定存在促销信息传播不广泛、商家线下资源闲置无法充分利用的问题。在新模式发展不明朗，资本市场过度解读推进的情况下，O2O模式得到了井喷式的快速扩展应用，例如O2O美甲、生鲜等等。当然这和智能移动设备的普及、[移动支付](http://baike.baidu.com/view/10750148.htm)的成熟、大数据广泛应用等社会元素的发展是密不可分的，移动互联网的发展使得O2O服务的使用频率和粘滞性不断提高，O2O不断融入日常并不断改善社会的日常。不过这一时期由于资本市场的过分解读和炒热使得这一时期移动互联网行业泡沫化异常严重。

3、O2O 3.0

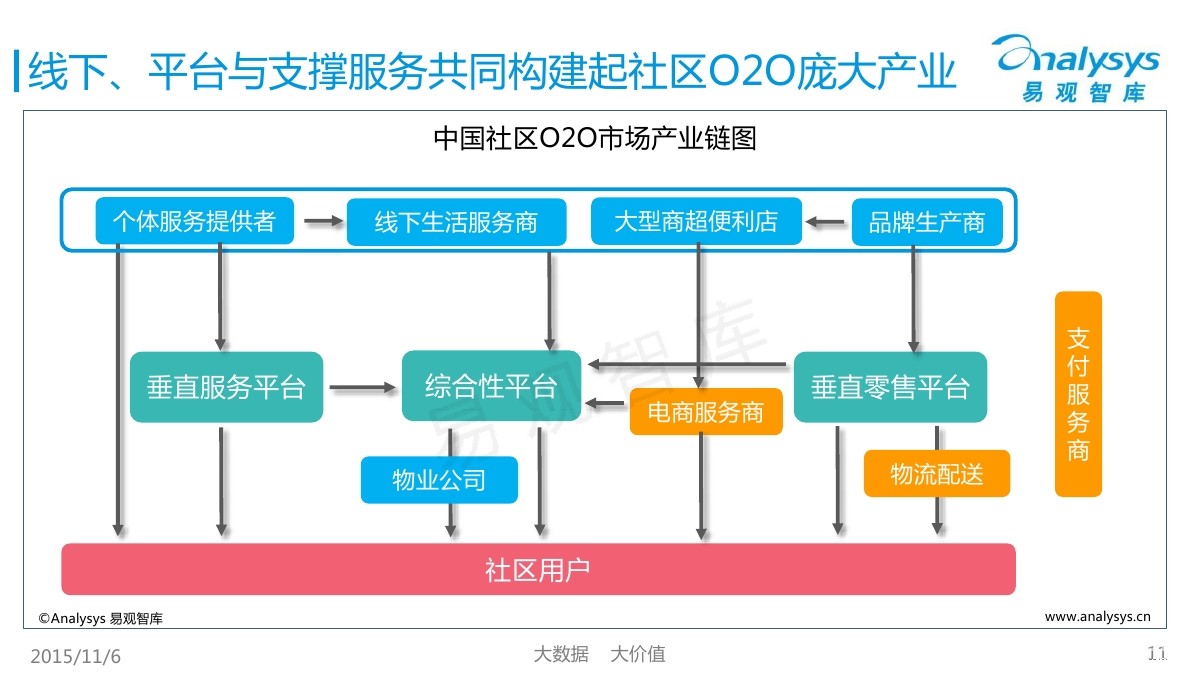
本质上3.0时代是2.0时代充分发展的一个必然阶段。该时期O2O模式的应用开始不断细化，垂直深化，而且这一时期某一领域的佼佼者往往能称为该领域的独角兽企业。如嘀嘀打车、Uber、snapchat等等。久而久之当这些独角兽在垂直细分领域充分发展之后就会开始新的平台化模式发展。由开始的垂直细分领开始不断地横向扩张，触及全行业。例如[饿了么](http://baike.baidu.com/view/2244685.htm)开始只是做外卖平台，之后就开始吸收第三方物流团队。他们以加盟商为主，以自营配送为中心，通过大量外包合作实现复杂订单的低成本配送。配送商品类型不断扩展至生鲜、商超等产品，做到全面的平台化的大整合经营。下面将用图1. 2对当前中国O2O市场的发展状况做一个全面性概括。

图1.2 中国社区O2O市场产业链图

## 1.3 论文研究的意义

从当前的O2O商业模式与传统计算机技术和互联网的结合发展可以看到了，首先O2O模式是对人类社会的发展起到极大推动性作用的商业模式，这已经证明O2O模式是无可怀疑的成功商业模式。这让人们也看到互联网依然有巨大的潜力有待人们继续深入挖掘。由现有的O2O已经可以看到O2O在传统的日常消费如衣、食、住、行中应用更为合适。我们现在所能看到的打车、外卖、住宿等O2O应用所解决的需求痛点也都基于的生活需求。

同样本论文要实现的系统也是用O2O模式改善人们日常工作学习中常用的文件打印这一服务。通过调查发现当前打印这一行业与出行打车、订餐有着极高的相似性，例如同样需要向下体验，同样存在商家与普通用户之间的信息匹配度较高的问题。另外还存在一些特有问题这里给出一个简单的列举

（1）用户去线下打印时需要自带存储设备这时就会存在设备可能忘记带的情况。

（2）可能存在设备使用完之后忘记拔走的情况。

（3）设备来回插拔打印社的电脑可能插口不足需要排队。

（4）各种移动设备集中在同一PC上插拔使用可能会导致中毒后交叉感染。

（5）校园用户由于日常作息时间的相似性较高从而出现比较集中的打印需求，这个时候就需要长时间排队打印、取文件等，时间利用率较高。

（6）由于普通用户的打印需求存在时间集中，这也导致了商家的工作时间受到用户时间的影响，存在忙时忙不过来，闲时无所事事和设备闲置。

（7）在打印业务繁忙时经常存在用户的打印需求商家理解错误的情况，这也就给商家和用户都造成了不必要的时间和资源浪费。

通过开发本系统就可以合理解决当前商家和普通用户所面临的此类问题。首先，通过系统可以使得普通用户直接通过互联网在线就可以下单进行打印，无需亲自上门就可将需要打印的文档传至商家端，如果商家收到之后刚好设备处于闲置状态，那么他就可以选择立即打印，当然反之商家可以选择将大量预定订单统一归置好之后再进行有序的打印。这样就可以使得原本繁琐且易漏的打印工作变得井然有序并且可以根据时间合理分配了。其次，这样的远程传输不再要求打印者必须携带自己的存储设备亲自上门打印，降低了打印者因为忘记携带设备而耽误打印的风险。同时也避免了打印社电脑USB接口不足而浪费的非必要排队时间。最后无论是日常打印需求比较大的个人用户，还是大印任务量较大的商家，都有相当规模的打印历史需要管理，以便以后查阅参考。这时在线打印系统就很好的做到过往打印历史的管理省去了繁琐的手动笔记过程。

通过前面的分析可以发现在线O2O模式是势不可挡的新模式，另外分析当前的打印商铺的交易流程以及个人用户的打印需求，也可以发现O2O模式直接复制到在线打印领域基本就是对原有生产力的一次大幅度提升。这种提升既提升了整体的工作效率而且也是对每一环节，每一个交易角色的工作优化。

本论文就将直接撰述这种在线打印交易系统的设计以及具体实现，用一个完善成熟的设计和开发流程将本课题做一个完整的展现。

## 1.4 论文的组织结构

本文主要概述作者设计开发的一款基于O2O模式的在线打印系统。通过后面的几个章节依次展开叙述本课题地设计思路和开发过程中的具体实现方法与技术。同时也对设计开发中遇到的一些问题和个人的一些思考进行了记录。接下来就对论文的整体结果进行概括性的介绍。

第1章是绪论，首先介绍了本课题的研究背景，从O2O模式的产生一直到移动互联网蓬勃发展的今天O2O模式的新定义。继而介绍了O2O在国内外发展的现状、对当今比较热门的几家O2O模式的商业公司进行了分析评论。最后就是对本文的整体结构进行概括性的介绍。

第2章是系统开发环境及相关技术的介绍，首先从整体上明确该系统的开发目的，完成开发所用到的后台数据库，以及开发过程中用到的语言。介绍后台数据库管理系统和开发语言的特点。最后介绍了相关技术，这些技术都将在系统的开发过程中用到。

第3章是在线打印系统的分析设计，主要包括数据库的设计和模块设计。首先进行系统的需求分析以及对系统功能模块的划分，再进行数据库的设计和建立，最后对各个模块进行设计，画出它们的流程图，为系统的开发做准备。

第4章是在线系统的实现，包括访问数据库的实现和各个模块的实现。首先介绍访问数据库的实现，然后根据前一章设计的流程图，用Python和PHP语言实现各个模块。

最后是本课题的结论部分，这部分是对本课题进行的整体性的概括总结，同时也记录了一些工作中的不足和需要改进的问题。

# 第2章 系统开发环境及相关的技术

本章主要介绍基于O2O模式的在线打印系统设计和开发的环境，以及这其中用到的相关技术。大部分开发环境是以软件工具和集成环境组成。相关技术则是指系统开发中使用到的开发技术，在系统设计与实现内容展开之前对相关的技术做一定的介绍。

## 2.1 开发环境概述

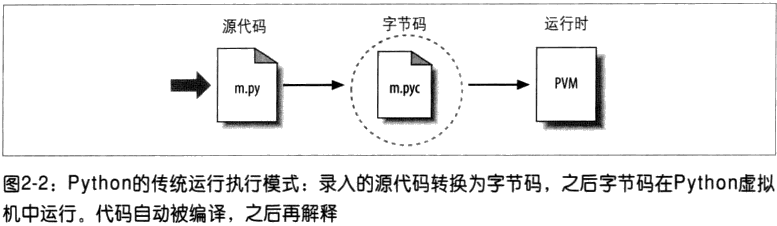
本系统是基于B/S架构的软件系统。本课题原本计划开发的系统本身对网络存储管理以及对在线文件传输管理、消息传递的时效性都有较高的要求。再者结合当前的计算机技术发展以及本项目的需求设计，可以有效而准确的选择适合本项目设计开发的环境和工具。

作为一个完善的B/S架构的软件系统，一个方便合适的数据库管理是必不可少的[21]。这里提到了合适，这就需要真正的根据系统的设计以及预期的应用场景选择合适的数据库管理系统[22]。本系统对出存储的规模需求不高，但对数据库系统的灵活性和各个计算机系统平台的兼容性需求较高，因此考虑到这些，本系统选择用轻量级的开源数据库系统Sqlite。另外选择数据库系统类型也受到了本系统采用的一些开源的系统架构的影响，因此本系统的另外一些模块也会采用其他类型的数据库管理系统。这一部分会在后面详细阐述。

其次本系统所用到的开发环境，开发环境可以分为开发工具和开发所用到的框架结构、开发语言等。本项目主要用到了html、css、javascript、php[16]、python、sqlite、mysql开发语言，由此选定了Sublime\_text.3、Zendstudio、Pycharm.3、navicat。首先html、css、javascript这是B/S系统前端界面必备的三大开发语言。其次本系统后台语言选择了php由于php特别适合小型论坛[17]、企业商城的搭建，因此本课题选择了用php语言作文论坛模块的编写实现。在线存储模块的后台框架使用的是Django框架因此需要用到Python语言进行编写实现。

2.1.1 Python语言简介

Python语言是一种面向对象的计算机编程语言，这是一门开源的跨平台的编程语言，同时这门语言又是面向对象的[2]。和java类似Python也是面向对象的，解释型的编程语言[3]。关于解释型编程语言的运行原理在图2.1中做出解释。

图2.1 Python的运行执行模式图

Python具备现代编程语言所应具有的一切功能，Python具有非常丰富的标准库，同时经过多年的发展Python也已经有了十分丰富的带三方模块[3]。

Python语言特别易于使用，Python语言更接近英语，无论是创建过程还是运行过程都十分的迅速，因此python可以使得开发者将大部分的注意力集中在开发程序本身[4]。与java和C#不同的是python也是面向对象的但是python却不是必须OOP的[5]。这就意味着同样运行小程序，python却不需要特别复杂的额外的代码部分。另外python不仅共能强大而且灵活性也特别好。

Python还被开发者戏称为“胶水语言”因为python可以和C、C++、Java等语言结合使用，这也就意味着在开发者已经有了其他语言所开发的项目的情况下他可以直接在此基础之上附加上一些python程序之后就可以运行一个完整的程序了[6]。然而这还不是最值得称道的地方，python可以在现有的任何计算机系统上运行，如，OSX、Linux、windows。毫不客气的说只要是能运行python解释器的平台python都无往不胜[7]。最后不得不提的是python语言也是开源的。

2.1.2 Sqlite简介

Sqlite，是一种轻量级的数据库，Sqlite本来是为嵌入式设备而设计出来数据库。因此这种数据库足够小，同时Sqlite对内存的需求量也是极低的几百KB而已。与python类似Sqlite也支持多种计算机系统平台。

在事务处理方面，SQLite通过数据库级上的独占性和共享锁来实现独立事务处理。这意味着多个进程可以在同一时间从同一数据库读取数据，但只有一个可以写入数据。在某个进程或线程向数据库执行写操作之前，必须获得独占锁。在获得独占锁之后，其他的读或写操作将不会再发生。

SQLite数据可采用的数据类型为动态数据类型，例如当某个值要存入到数据库时，SQLite会首先对它的类型进行检查，如果该数值的类型与其所要插入的列的数值类型不匹配，SQLite则会将该值转换成与该列相同的类型，如果无法转换的话，Sqlite就会将该值以其原来的数据类型直接进行存储，而这种数值；类型在SQLite的官方文档中被称这为“弱类型”。

2.1.3 Pycharm.3、Sublime\_text.3简介

PyCharm是一款知名的Python IDE，Pycharm既具有其他IDE一样所具有的编程辅助功能如，代码补全、代码片段、智能提示等等。同时具有最丰富的Python语言开发时用到的功能，例如Python重构、支持Django框架下的专业的web开发。为此Pycharm自带HTML[18]、CSS、Javascript[19]此类前端语言编辑器。这样的话开发者就可以通过Pycharm一款编辑器完成完整的Django专业Web项目开发工作。

Sublime\_text.3是一个十分优秀的代码编辑器，这款编辑器是于2008年由程序员Jon Skinner开发出来，Sublime\_text.3具有良好的易用性和强大的可扩展性，易用性在于Sublime\_text.3具有智能的语法高亮，代码自动完成功能。Sublime\_text.3强大的扩展能力另起拥有很多丰富的第三方功能插件。优秀的插件如，GBK to UTF8将文件编码从GBK转换成UTF8，快捷键Ctrl+Shift+C、zenCoding、zenCoding是一种快速的html、css编写方式。默认用tab键.已经更名为Emmet、JS Format一个JS代码格式化插件。默认ctrl+alt+f BracketHighlighter括弧高亮插件，清晰明了Git代码版本管理、Chinese Localization 汉化插件，支持无缝切换中文、日语、英文。

## 2.2 相关技术简介

这里要说到的是本系统从设计到实现过程中，运用到的一些主要技术及架构。这里做这些简要的技术说明是为更好的理解后面的章节中的相关开发概念和术语，同时做一个相关技术的简介也是将本系统设计的合理性以及实现过程中的一些运用方式做一个展现，以增强读者对系统设计实现合理性的理解。

2.2.1 MVC与MTV模式

MVC模式的全称是（Model-View-Controller）翻译后是model(模型)-view(视图)-controller(控制器)，这样就可以将一个应用分成三层分别是模型层、视图层、控制层[10]。同时当前MVC模式是一种目前在计算机软件开发领域比交成熟且常用的开发模式。下面的内容将对MVC模式做一个简要的介绍。

1、视图

视图(View)指用户交互的UI界面[11]，在Web应用中界面可以概括为前端界面。现代的web应用的复杂性和规模性都在不断的提高，因此web应用的操作界面开发设计也变的越来越复杂。一个成熟的web应用一般据有许多视图，也就是一般所称的界面，在MVC模式中视图处理指的是对视图上的数据的采集和处理，但是并不包括对界面上的业务流程的处理。业务流程部分的处理其实是由模型(Model)部分来处理的[12]。可以举例说明一下，在线交易系统中用户的交易记录的视图就是接受来自(Model)模型的数据将这些数据输出显示在历史记录页面中来，同时还要将用户从视图界面中输入的数据和发出的请求传送给控制模块和模型模块。

2、模型

模型(Model)用来将业务的流程、状态进行处理。这一部分对业务流程的处理相对于其它的模块来说是透明的，模型从视图中获得业务请求的数据，获取数据后通过自身预设的业务规则执行并输出要呈现给视图模块的结果数据。对于MVC模式最重要的核心部分就是业务模型的设计。当把这三大模块的标准界定清楚之后，开发人员真正需要做的，就是设计出满足需求的业务模型。通过MVC模式可以看出，良好可用的模型就是把应用的模型准确的分离出来，同时不同的开发模型的差异也就体现在模型各个层次划分的优劣上。通常在划分系统设计模型时抽象的概念与具体要模拟的实体要控制在合理的“距离之内”不能过于抽象而背弃了实际也不能过于要求接近实际。MVC不需要开发者特别了解模型的设计方法，作为开发者只需要懂得怎么组织管理各个模型块就可以了。这里可以用面向对象的编程方式类比MVC模式，这就好比MVC定义了一个最高级别的类，要求的子类只能做哪些，但并不限制只能做这些。模型模块中中还有一个很重要的类型就是数据模型。数据模型主要指实体对象的数据保存模式。我们可以将这个模型单独列出，所有有关数据库的操作只限制在该模型中。

3、控制

控制(Controller)的作用就是在从用户接受请求后，通过这些请求将模型与视图进行逻辑匹配，最终在经过模型与视图的处理之后将请求的处理结果返回给用户。同时在MVC模型中控制模块的划分也是十分的清晰，控制模块的存在正如一个中转器，它在接收到用户的请求之后，准确的为每一次请求选择对应的模型和视图。由此也可以明确的发现控制层并不会对用户的请求做任何的改变而仅仅只是中转，用户的请求在控制层的进出前后也不会有任何的变化。简单举例，当用户点击过某一个一个链接之后，首先由控制层接收到这个请求，但是控制层并不对该信息做任何的处理，控制层准确的将信息传递给对应的模型，并同时将处理方式交给该模型，在模型将数据完整的处理过之后，控制层再选择符合要求的视图返回给用户。这时就能发现，模型与视图之间是多对多的关系。例如，在当用户通过某一个视图的控制器改变了其对应的模型数据，则此时视图层所有依赖该模型数据的视图都将随着这一变化而发生变化。因此无论什么时候只要发生了模型数据的变化，控制层都会将这些变化反映至所有与之相关的视图上。这就是一种模型的变化-传播机制。下面将通过Model、View、Controller三者之间的关系和各自的主要功能对MVC模式的原理进行介绍，如图2.1所示。

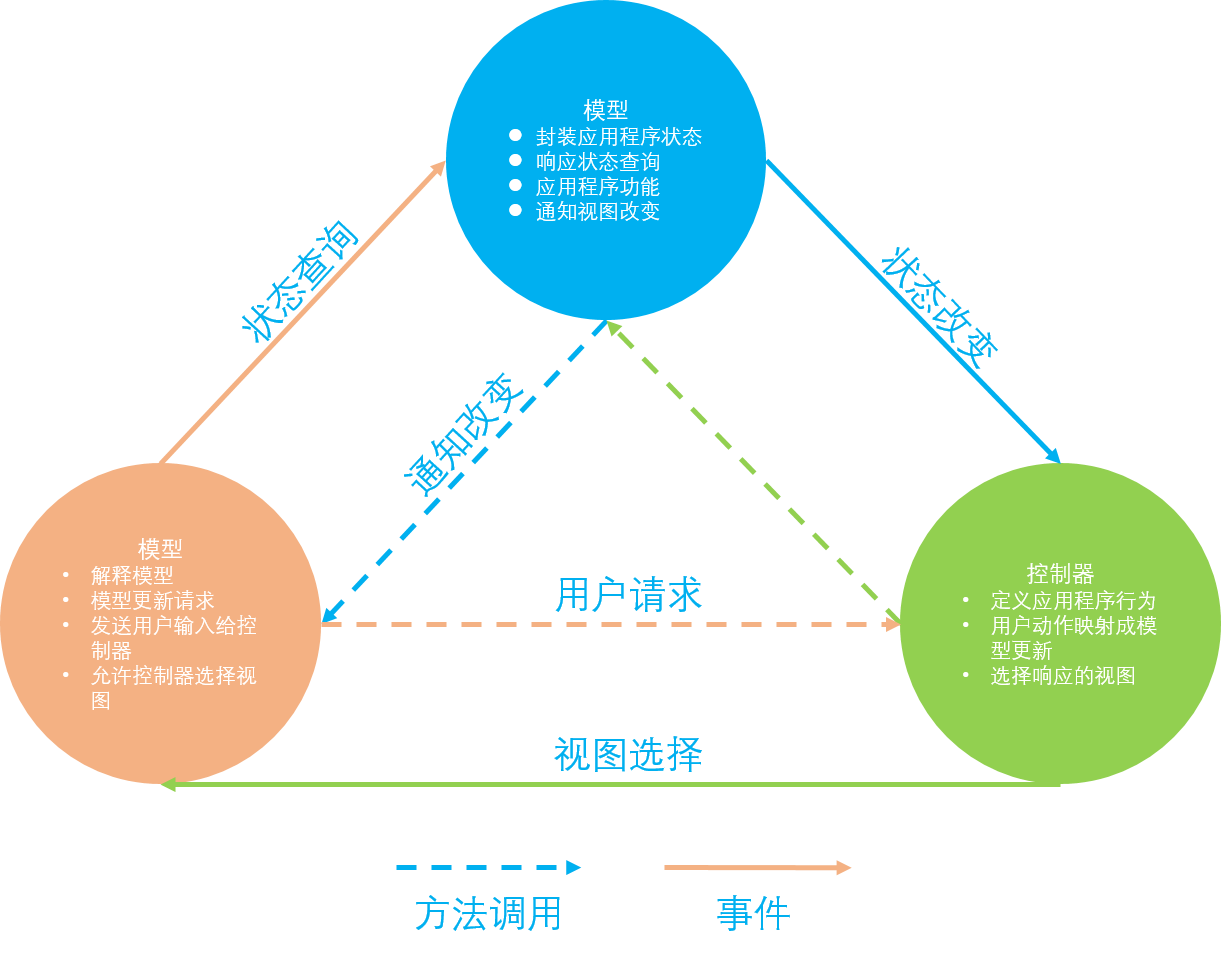


图2.2 MVC开发模式逻辑图

MTV模式的全称是（Model-Template-View）翻译后是model(模型)-Template(模板)-View(视图)[13]。此处的 Model（模型）与 MVC 中的 Model（模型）相同，是数据访问层，它包含了关于数据处理的一切内容，例如，如何获取数据，如何验证数据，它具有什么行为以及数据之间的关系，写在 models.py 中。

MVC中的 Controller（控制器）由 URLconf 单独来实现，写在 urls.py 中，MVC 中的 View（视图）在 MTV 中则被分成了 Templates（模板）和 View（视图控制器）两部分。Templates（模板）相当于 MVC 中的 View（视图），它包含了呈现相关的决策，如内容怎样在Web页面中显示以及其它类型的文档 ，写在 html 文件中。View（视图控制器）决定了要呈现哪些数据，以及如何去渲染 Templates（模板），写在 views.py 文件中。View（视图控制器）相当于一座桥梁，连接了 Model（模型）和Templates（模板）[14]。接下来将通过模型、模板、视图的相互关系具体的对MTV模型的原理进行概述，下图是图2.2所示。

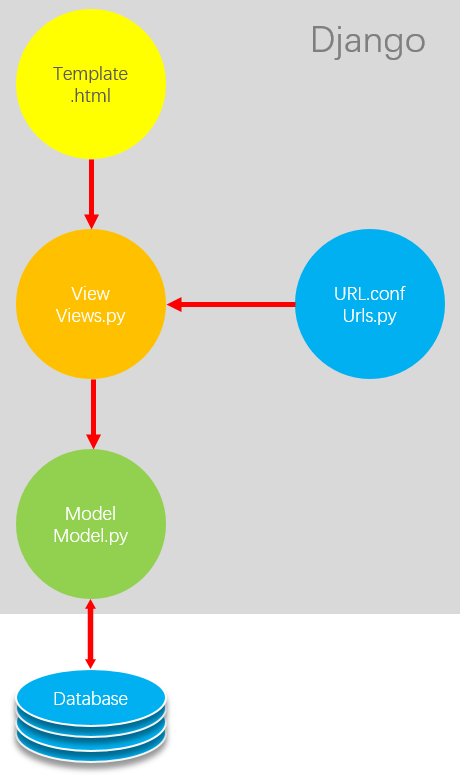


图2.3 MTV模式原理图

2.2.2 Django简介

Django是一个开放源代码的Web应用框架，是由Python编程语言写成。采用了MVC和MTV的软件设计模式，即模型M，视图V和控制器C[8]。这也正是为什么本课题在开始这一部分之前就将Python语言和MVC、MTV开发模式进行了详细的介绍。另外前面所提到的Sqlite数据库也是Django框架原生匹配的数据库，在此之前本文也做了准确的介绍。

在Django中，控制器接受用户输入的部分由框架自行处理，所以 Django 里更关注的是模型（Model）、模板(Template)和视图（Views），称为 MTV模式[9]。它们各自的职责如下。

1、模型（Model）

即数据存取层 处理与数据相关的所有事务，如何存取、如何验证有效性、包含哪些行为以及数据之间的关系等[10]。

2、模板(Template)

即表现层 处理与表现相关的决定，如何在页面或其他类型文档中进行显示[11]。

3、视图（View）

即业务逻辑层 存取模型及调取恰当模板的相关逻辑。模型与模板之间的桥梁[11]。

1、系统结构

 Django 视图不处理用户输入，而仅仅决定要展现哪些数据给用户，而Django 模板 仅仅决定如何展现Django视图指定的数据[12]。或者说, Django将MVC中的视图进一步分解为 Django视图 和 Django模板两个部分，分别决定 “展现哪些数据” 和 “如何展现”，使得Django的模板可以根据需要随时替换，而不限制于内置的模板[13]。 至于MVC控制器部分，由Django框架的URLconf来实现。URLconf机制是使用正则表达式匹配URL，然后调用合适的Python函数。框架把控制层给封装了，无非与数据交互这层都是数据库表的读、写、删除、更新的操作。

2、设计优势

Django本身被设计之处的主要是为了简便、快速的开发数据库型的网站。它强调代码复用,Django中许多多组件都可以以“插件”形式应用于整个框架，同时Django有许多功能丰富的可扩展插件，开发人员也可以开发出自己的需要的工具包[14]。因此Django具有较强的可扩展性。这也是它之所以强调快速开发和DRY(Do Not Repeat Yourself)原则的良好提现。

（1）对象关系映射

以Python类形式定义数据模型，ORM将模型与关系数据库连接起来，将得到一个非常容易使用的数据库API，当然也可以在Django中使用原始的SQL语句。

（2）URL 分派

通过正则表达式匹配URL，开发者可以设计任意的URL，而且并没有框架的特定限定。这正如所有开发者期待的一样灵活

（3）模板系统

Django强大而可扩展的模板语言，可以分隔设计、内容和Python代码[15]。并且具有可继承性。

（4）表单处理

开发者能够快速的生成各种表单模型，实现表单有效性的检验。而且可以简单的自定义的模型实例中生成需要的表单。

（5）会话

用户登录与权限检查，快速开发用户会话功能。同时Django内置国际化系统，可以便捷的开发出多种语言的网站[14]。

（6）自动化的管理界面

不需要开发者投入过多的精力来创建后台管理和更新内容。Django自带一个ADMIN site，类似于内容管理系统。

3、工作原理

首先用manage .py runserver 启动Django服务器时就载入了在同一目录下的settings .py。该文件包含了项目中的配置信息，如前面讲的URLConf等，其中最重要的配置就是ROOT\_URLCONF，它告诉Django哪个Python模块应该用作本站的URLConf，默认的是urls .py 。当访问url的时候，Django会根据ROOT\_URLCONF的设置来装载URLConf。 然后按顺序逐个匹配URLConf里的URLpatterns。如果找到则会调用相关联视图函数，并把HttpRequest对象作为第一个参数(通常是request)。最后该view函数负责返回一个HttpResponse对象。

## 2.3 本章小结

本章首先根据在线打印系统的功能需求，选择了合适的开发框架，并就此开发过程所需要的开发序言、过程模式、数据库、开发环境进行了初步的介绍。这为后面章节中的系统设计、开发做好了充足的准备。

# 第3章 在线打印系统的分析设计

## 3.1 系统需求分析

需求分析是实施设计的一个基础，确定系统的设计任务和设计目标，才能很好地设计解决方案。在软件生命周期中，需求分析是很重要的一个阶段。需求分析的质量对软件开发的影响是深远的、全局性的，高质量需求需求分析对软件开发往往起到事半功倍的效果。[软件需求](http://baike.baidu.com/view/10192.htm)分析主要以下三个方面的分析：

1、业务需求分析

业务需求分析（business requirement），这一部分反映的是相关组织机构或用户对系统、产品层次的目标要求，它们在项目视图与范围文档中予以说明。在本课题中业务需求更多的是关注本系统究竟能给普通的拥有打印需求的用户、和另一端的商家带来多少的工作效率提升，同时作为一个纯粹的学术性研究课题，本系统不需过多考虑系统开发后的经济效益如何。

2、用户需求分析

其次是用户需求分析(user requirement) ，这一个层次关注的是用户在使用系统过程中需要执行的工作任务。

3、功能需求分析（也包括非功能需求分析）。

最后是功能需求分析(functional requirement)这部分用以指导开发人员需要实现的软件功能，使得用户能利用这些软件功能完成他们的工作，从而满足了组织内提出的业务需求。

本系统从软件结构分析主要包括前端页面、后台逻辑框架、以及数据库部分。首先对于数据库的设计一定要建立满足数据一致性和完整性的稳定的数据关系库。而这也正是设计开发一款真正能解决用户需求的软件系统的 基础。另外前端页面的功能布局设计一定要符合人机交互的使用习惯，同时在界面风格上也要不断接近现代化的设计风格。最后后台逻辑框架也要满足软件开发中的高效性、稳定性、健壮性。

此在线打印系统，目标就是降低普通用户打印过程中不必要的时间浪费和繁琐的文件管理，同时均匀化分配线下打印商家的打印高峰和打印空闲时间的任务量。因此本系统首先建立普通用户的注册登录模块，在普通用户登录之后，首先进入的就是个人网络资源管理平台，在这里普通用户可以上传自己需要打印的文件，用户还可以进行个人资料编辑管理。对个人网络空间内的文件进行增删改查，同时文件的管理还包括文件目录结构的的更新修改以及文件的重命名。以上即为文件管理部分可实现的功能。文件在上传完毕之后就可以进入打印流程，通过云打印用户可以对要打印的文件进行打印前的设置，具体设置项如打印张数、打印的颜色、用户喜欢的商家等等，而这时系统也会根据商家预设的单价以及优惠方式自动计算出。商家在收到普通用户发来的打印订单请求之后就可以进行订单确认以及选择立即或者稍后再将用户订单中的文件下载至本地。

同时在商家和个人用户的管理界面中都可以对过往的打印历史订单进行查看和管理，例如选择对历史订单再打印一份或者删除某一条或某几条历史记录。另外与订单流程无法分离的就是商家与用户之间的积分系统。积分系统可以接受普通用户的充值，支付过程中积分能根据订单的消费额自动减少积分额度，另外商家部分也可以对盈利积分进行提现。

通过分析，大致可以将整个系统开发分成以下几个步骤进行开发。

（1）首先建立数据库，通过数据库建立用户注册信息表，存储普通用户信息和商家注册信息。数据库使用Sqlite数据库。

（2）通过细化系统结构，将本课题划分出各个功能模块，并且根据功能模块逻辑结构设计出程序流程图。

（3）开发系统功能界面，使用HTML、CSS、Javascript语言进行前端页面的编写，同时保留与后台逻辑架构的接口代码[20]。

（4）按照流程图的指导，进行开发程序的部署安装。由于使用了Django框架因此连接数据库的操作需要按照Django框架的框架内部的规范链接。

（5）进行功能逻辑部分的编码工作，并且完成所要开发的功能。整个系统正是需要该部分进行业务处理，因此该部分可以说是整个系统的逻辑控制中心。这部分的工作量也是最大的。

（6）测试，初步开发完成之后就需要对整个系统进行全面的检查测试，检查其功能合理性，测试系统中的Bug。

## 3.2 功能模块的划分

结合系统设计的要求，再依据上面的软件工程开发原则，可以将系统划分为一下七个模块。分别为用户登录模块、商家登录模块、文件管理模块、在线交易商城、打印前设置、商家后台模块、资源交流论坛、由此七个模块得出系统的总体框架图。如图3.1所示。

基于O2O模式的在线打印系统

用户登录模块

在线交易商城

文件管理模块

商家登录模块

打印前设置

商家后台模块

资源交流论坛

图3.1 系统总体框架图



图3.2 系统功能结构图

可以从系统总体框架图中看出，各个模块之间有着一定的依存关系，但大体上分为七大模块。

1、用户登录模块

对用户名、密码、验证码进行验证，若是有效的用户名密码则可以立即登陆成功，若该账户不存在则提醒用户确认用户名密码，若系统根本不存在该账户则提醒用户注册新的账号，这时用户可以通过注册链接进入注册页面。

2、商家登录模块

对用普通商家用户名、密码进行验证，若是有效的用户名密码则可以立即登陆成功，若该账户不存在则提醒用户确认用户名密码，若系统根本不存在该账户则提醒商家用户注册新的账号，这时商家用户可以通过注册链接进入注册页面。

3、文件管理模块

是整个系统设计的重点、难点，主要完成对普通用户网盘内文件的增、删、改和查功能。良好的网盘文件管理系统应该给用户提供常用的操作系统资源管理器一样的使用习惯。网盘文件管理模块可以对文件进行上传、下载、重命名、进入打印流程等。

4、在线交易商城

系统管理员作为平台方运营方提供一个常用学习资料的在线销售平台。

5、打印前设置

这一模块与文件管理模块联系较为紧密，在用户使用网盘文件管理的云打印功能时，文件打印就会进入打印前设置流程。这里用户可以根据个人的打印需求对打印文件进行设置，如打印页数、纸张类型等。

6、商家后台管理模块

商家用户通过后台管理模块可以进行商品发布、编辑、删除、以及订单管理。

7、资源交流论坛

所有用户可以在该部分进行资源分享，同时可以在论坛内部创建资源交流小组，小组内进行相关资源话题讨论。

## 3.3 数据库设计

3.3.1 数据库的设计

本系统是完善的B/S架构的系统，具有稳定的逻辑架构，良好的交互界面。同时也具有一个稳定合理的数据库。稳定合理的数据库系统不但需要满足系统开发中的客观需求，同时也要选择合理的数据库类型，在设计数据库是才能完美的与系统功能进行结合。

接下来将对这一点进行深入的介绍，数据库设计的定义是针对某一系统环境，选择最佳的数据库模式，创建合理的数据库及数据库管理系统，使数据库达到最良好的数据存储效果，从而满足项目开发需求。要做好完善的数据库系统设计，应该首先明确系统功能要求。以此对数据库进行需求分析。基本的分析思路是，首先对要处理的对象进行分析，进而对系统中的各个模块进行分析，拆解其中的数据需求。再次对用户需求进行分析，将用户需求转化成数据库可描述的数据模型。通过对这些用户数据模型的抽象、综合、归纳，提炼出一个完整的DBMS概念模型。最后将概念模型转化为关系模型，至此就就可以通过DBMS系统进行数据库的实现操作了。

这里首先给出合理的系统用例图，作者根据系统中各类用户的功能需求，以及结合了系统可行性给出如下用例图。图3.1给出系统中各类人员的用例关系。图3.2给出系

统中系统的ER图。

图3.1 系统用例图

图3.2 系统的ER图

在数据库设计过程中，选择一种合适的数据库模式对之后数据库的稳定性，流畅性十分重要。这也在本课题选择以sqlite为数据库的考虑之内。关系数据库设计的目标就是要从各种可能的关系模式组合中选取一组关系模式来构成一个数据库模式，使得既不必存储不必要的重复信息，又可以方便地获取信息。

接下来，本章将根据上节得到的功能模块对数据库进行设计。在普通用户登录模块和普通商家管理模块中，就需要根据当前登录系统的用户类型的不同进行不同的管理，这就需要对用户账户数据进行存储，因此需要创建一张普通用户信息表，和一张商家账户信息表。在文件管理模块中需要对存储的文件进行索引信息管理因此需要一张文件记录表。在商家发布了商品信息之后需要对所有商品信息进行记录因此需要一张商品信息表。商家用户需要接收来自普通用户的文件打印订单，因此需要一张商家交易信息记录变。

最后统计一下，该系统中需要在数据库中建立11张表。用户、商家、文件、订单、商品分类、商品、店员、评价、以及店员和商店的多对多关系映射表，同时auth\_user和django\_session也是流程中的一部分，其中auth\_user用于存储用户的基本信息，这个是商家和用户共享的，django\_session是用于程序的Session管理。下面将对每一张表内的字段名称、字段类型及字段含义进行说明。

首先是用户信息表，无论是普通用户还是商家用户，登录系统时，系统需要对用户名和密码进行数据库查询以判断其合法性，因此在用户信息表和商家信息表里都需要设置用户名和密码字段。另外由于本系统存在两种不同的注册账户类型，因此从数据库设计的角度应该对普通用户账号和商家用户账号进行登陆验证以此来作为登录验证通过之后返回何种页面类型的依据。但在本系统中为了省去不必要的判断过程，并且能给用户以清晰的系统逻辑思路。因此在系统中为普通用户和商家用户预留了不同的登录入口，进而在数据库系统中就可以为商家和普通用户建立不同的用户信息表项。表中的各个字段如表3.1所示。

表3.1 用户信息表

|  |  |  |
| --- | --- | --- |
| 字段名称 | 字段类型 | 字段说明 |
| name | TEXT | 用户名，不能为空 |
| icon | TEXT | 用户头像，不能为空 |
| tel | TEXT | 用户电话 |
| credit | REAL | 积分 |
| address | TEXT | 地址 |
| resRoot\_id | INTEGER | 是用户的文件根，也就相当于Linux下的/ |
| user\_id | INTEGER | 用户编号，是用户信息表的主键 |

商家的信息与个人用户的信息资料相对接近 因此相应的数据项也比较相似，因此商家这里也有用户名、密码、电话、头像、id等表项。除此之外商家账户与个人账户的差异在于，商家账户里的积分是可以提现的，而个人账户则不能，因此商家账户的信息表里有一项是商家账号绑定的支付宝账号的字段。另外用户商家但是和个人用户有一点最大的不同是，商家账户有一项shop\_id数据项，也因为每一个商家用户都对应有自己的店铺因此shop\_id是商家账户与店铺相关联的唯一键值。另外由于一个商家可能存在多个店铺因此一个商家账户和店铺之间的关系是一对多的关系。表中的各个字段如表3.2所示。

表3.2 商家信息表

|  |  |  |
| --- | --- | --- |
| 字段名称 | 字段类型 | 字段说明 |
| 字段名称 | 字段类型 | 字段说明 |
| user\_id | INTEGER | 用户编号，是用户信息表的主键 |
| name | TEXT | 用户名，不能为空 |
| icon | TEXT | 用户密码，不能为空 |
| tel | TEXT | 店员联系方式 |
| shop\_id | INTEGER | 店铺编号 |

店铺信息列表里记录的都是每一个店铺的基本信息，在系统里除了描述店铺对象的一些基本信息，需要重点说明的是因为店铺也需要与普通用户以及商家之间发生积分兑换，因此店铺对象也具有credit表项可以与商家和用户之间的credit表项产生数据转移。另外店铺的积分属于系统内自的虚拟货币，商业性交易必然要产生现实的现金交易，因此本系统为店铺对象设置了与支付宝账户表项。用以存储与店铺账户绑定的支付宝账户，进过后台的积分兑换算法就可以实现积分提现了。店铺信息表如表3.3所示。

订单表项，是用来记录打印文件订单的表项，订单表是数据项最多的表。它记录了订单的id和下单日期以及对应得店铺信息，如店铺位置、名称、联系方式、订单对应的积分金额、点单中对应的文件、订单当前所处的状态等等，另外很重要的是订单里还记录着下单流程中所有的设置项如下单的用户、收获地址、联系方式、打印的颜色、页数 、纸张类型等等。表中的各个字段如表3.4所示。

产品信息表中记录的是商家所发布的商品的详情，如商品id、名称、文件id、价格、文件打印的颜色、纸张大小等这些都是打印文件时大家所关注到的文件属性。表中的各个字段如表3.5所示。

表3.3 店铺信息表

|  |  |  |
| --- | --- | --- |
| 字段名称 | 字段类型 | 字段说明 |
| resROOT\_id | INTEGER | 文件根目录类似于linux下的/ |
| name | TEXT | 店铺名称 |
| icon | TEXT | 店铺logo |
| tel | TEXT | 店铺联系电话 |
| address | TEXT | 店铺地址 |
| credit | REAL | 店铺积分 |
| alipayAccont | TEXT | 绑定的支付宝账户 |
| alipayName | TEXT | 绑定的支付宝用户名 |

表3.4 订单信息表

|  |  |  |
| --- | --- | --- |
| 字段名称 | 字段类型 | 字段说明 |
| date | TEXT | 订单日期 |
| status | INTEGER | 订单状态 |
| shopName | TEXT | 店铺名称 |
| shopAddr | TEXT | 店铺地址 |
| res\_id | INTEGER | 要打印的文件id |
| user\_id | INTEGER | 打印者的id |
| credit | REAL | 订单所需积分 |
| num | INTEGER | 要打印的份数 |
| color | INTEGER | 要打印的颜色 |

表3.5 产品信息表

|  |  |  |
| --- | --- | --- |
| 字段名称 | 字段类型 | 字段说明 |
| name | TEXT | 产品名称 |
| res\_id | INTEGER | 文件id |
| price | REAL | 价格 |
| color | INTEGER | 打印的颜色 |
| size | INTEGER | 纸张大小 |
| name | TEXT | 产品名称 |

文件、文件夹信息表记录了文件的id、文件类型、文件名称、上传日期、分享状态、打印流程状态、下载与否状态、如果是文件夹的话则还有其子目录文件的信息表项。文件信息表有一点特殊之处是，由于文件信息表是与订单内的，表中的字段如表3.6所示。

表3.6 文件信息表

|  |  |  |
| --- | --- | --- |
| 字段名称 | 字段类型 | 字段说明 |
| type | INTEGER | 文件类型 |
| filename | TEXT | 文件名称 |
| date | TEXT | 上传日期 |
| file | TEXT | 文件自身 |
| share | INTEGER | 分享状态 |
| children | TEXT | 子文件 |

店员与商铺的映射表，把店员和商铺绑定起来。这里面记录的是店员id店铺、id以及一项用以判断某店员是否为某一店铺的拥有着的标志表项。各个字段如表3.7所示。

表3.7 试卷信息表

|  |  |  |
| --- | --- | --- |
| 字段名称 | 字段类型 | 字段说明 |
| seller\_id | INTEGER | 商家id |
| shop\_id | INTEGER | 店铺id |
| isOwner | INTEGER | 是否为拥有者 |

由于这几张表都有自己的主键，所以都满足实体完整性。这大大简化了数据在插入、删除、更新时的数据完整性等问题。

另外数据库中仍存在一些其他的数据表这里不再一一列举，主要原因是这些表项多数是Django框架运行所必要的预设数据表。如，有的表项是Django提供用以呈现基本的后台管理页面的等。另外还有一部分表项是在整个系统逻辑中关系型不强，仅仅是一些产品或者订单相关的属性数据项。

## 3.4 模块设计

这一部分是一个对整个系统进行合理划分的过程，将整个课题划分成几个主体大模块，再将大的模块划分出更易实现的小模块，本节就对模块的划分过程已经模块直接的逻辑流程关系进行分析。

3.4.1 登录模块的设计

这里说到的登录模块包括普通用户登录模块和商家登录模块。登录模块主要是一个用户输入用户名密码，程序收集用户输入后进行查询验证的过程。对于用户的输入需要首先验证其是否非空之后再对用户名、密码的合法性进行验证。合法性的验证又包括用户账号存在与否的验证，该判断过程一般通过查询用户表，验证是否存在这样一个用户。若用户名不存在则要求用户确认后重新输入。如果存则再进行进一步的验证判断用户输入的用户名和密码是否匹配。如果不匹配则要求用户重新输入密码。这样的登录验证设计正是防止非法用户的进入，影响平台的正常运营以及系统的安全性。其流程图如图3.1所示。

3.4.2 文件管理模块的设计

文件管理模块是所有模块的基础。只有文件管理模块的存在才能正常运行后面的文件打印流程、积分交换、订单系统才能接收到普通用户订单里要打印的文件，等等。文件管理模块包括文件上传、下载、打印、重命名、删除等。其中文件上传可以选择不同的文件夹目录。保存文件在不同的路径下。文件被上传至网盘之后还可以进行下载、删除、重命名等操作。如果单击文件夹可以进入文件夹，单击某一个文件则可以查看该文件详情，这些详情包括文件类型、上传日期、分享的次数、打印的次数等。文件管理模块功能结构图如图3.2所示。

图3.1 登录模块流程图



图3.2 文件管理模块功能结构图

在文件管理的流程末端，如果用户需要对点击过的文件进行打印，那么该文件将从文件管理模块进入文件打印前的流程设置模块。在文件打印前设置模块里，用户可以设置要打印的文档的页数、纸张类型、颜色等等。同时整个打印文件设置的过程也是本次打印订单生成的过程，包括填写打印文件收件人、联系方式、收件地址等等。打印前设置的流程图如图3.3所示。

图3.3 打印前设置流程图

3.4.3 商家后台模块的设计

商家后台模块相对来说是整个课题中复杂度相对最大的一部分，这里与其他模块都有较大的耦合关系。例如，首先商家后台通过订单流程与用户文件管理及用户下单流程相耦合。再者通过商品编辑发布功能又与商城模块相关联。另外商家后台模块通过积分系统与支付宝提现接口耦合。最后由于每注册一个打印店铺，每个店铺都可能有多个店员，此外店员中还有一位是店铺的管理员。因此综上分析店铺模块是整个系统中复杂度最大的一部分。商家后台的功能结构图如图3.4所示，订单管理模块流程图如图3.5。

图3.4 商家后台功能结构图

图3.5 订单模块管理流程图

在线商城模块的设计相对成熟，首先可以想象的是当前主流的在线交易网站的设计都已经相对固定。因此该部分的设计是在不改变用户在其他交易网站的用户体验习惯的基础上，针对本网站进行的个性化调整。首先，用户在网站上可以搜索自己需要的文件，整个交易首页上有清晰合理的相关资料分类，通过快速的分类信息查找，用户很快的定位到自己需要的资料，在了解完基本的商品信息之后就可以接下单购买。购买的过程中还可以针对自己的需求进行简单的需求设置，如颜色、纸张、份数等等。此流程图如图3.6所示。

图3.6 在线商城模块流程图

3.4.4 资源分享社区模块的设计

资源分享社区的设计，出发点是希望给所有的同学提供一个社区平台，大家在这里可以以发帖的形式分享自己的资源，通过自己独有的资源赚取其他同学的下载积分，这些积分可以在下载其他同学分享的资料时使用。另外在社区中大家可以根据自己不同的需求和兴趣点进入不同的分组，在小组里大家可以有针对性的讨论自己经常关注的兴趣话题，小组内大家也可以分享资源。作为社区运营人员平台方具有管理员权限，因此可以对普通用户的账号进行管理，同时也可以对帖子内容进行审核管理，包括删帖、关闭评论等。资源分享社区功能结构图如图3.7所示。资源分享社区资源下载流程图如图3.8所示。

图3.7 资源分享社区功能结构图

图3.8 资源分享社区资源下载流程图如

## 3.5 本章小结

本章主要对系统的数据库及各个模块划分进行了规划设计，这是系统开发实现之前十分重要的一步，只有有了正确良好的前期设计，后面的开发实现工作才能事半功倍。本章主要从系统的功能需求出发，划分出多个系统模块。系统模块的划分遵循软件工程所要求的高内聚低耦合的原则，以保证系统的高效性和稳定性。最后本章给出了各个模块开发设计的流程图，这为下面章节中的开发实现打好了良好的理论基础。

# 第4章 在线打印系统的实现

## 4.1 数据库的实现

上面一个章节已经对数据库的前期表项的规划进行了设计，基本确定了本课题所需要的数据表和数据表中用到的字段以及字段类型。本章节将会详细的描述数据库建立的过程。

Django框架中与数据库操作相关的程序都在models.py文件中，同时当前Django支持sqlite3, MySQL, PostgreSQL等数据库，而本课题所使用的数据库正是Django默认使用的sqlite3。与数据库相关的操作只需要在settings.py中配置数据库连接的设置项，接下来就在models.py代码中添加要创建的数据库各表即可。此外Django包含有丰富的API这在实际项目开发中是十分方便的。

由于Django无需通过直接编写SQL语句即可进行数据库操作。同时数据库的操作与Django项目的搭建过程密不可分，因此作者将从最原始的项目搭建过程开始描述数据库的连接创建过程。具体的Django操作步骤如下。

（1）django-admin.py startproject learn\_models # 该命令用以创建一个新项目。

（2）cd learn\_models # 通过该命令可以进入项目的根文件夹。

（3）django-admin.py startapp printer # 创建一个新的应用并将其命名为 printer。

这里需要提前对project 和 app之间的不同进行说明。最具有概括性的的区别描述就是一个是配置另一个是代码。一般同一个project里面可以有多个app以及对app的配置。project的作用主要是提供配置文件或者说提供一个规范配置的环境，比如如何定义数据库连接信息、当前已安装的app的列表、TEMPLATE\_DIRS模板路径等。每个app都是一套完整的Django功能集合，包括自由的模型和视图。

app的一个特点就是app可以很灵活的移植到其他的project中也可以被多个project同时使用。如果项目要求只需创建一个Web站点，这时候整个项目一般只需一个app就足以完成，如果要完成一个包含许多不相关的模块的复杂网站，那么整个项目可能需要把这些模块划分成不同的app，以便以后复用。

继续上面的项目步骤，将前面新建的应用（printer）添加到 settings.py 中的 INSTALLED\_APPS中，也就是告诉Django有这样一个应用。具体的操作步骤如下所示。

|  |
| --- |
| INSTALLED\_APPS = (  'django.contrib.admin',  'django.contrib.sessions',  'django.contrib.messages',  'django.contrib.staticfiles','printer',) |

这样在Project这个大框架之下就成功创建了我们需要的独立应用，这个应用的名字就是Printer。接下来就是数据库的连接测试，在此之前我们创建一个表用以在后面的链接调试时测试。首先打开 printer /models.py 文件，修改其中的代码如下。

|  |
| --- |
| from django.db import models  class user(models.Model):  name = models.CharField(max\_length=30)  tel = models.IntegerField() |

通过上面的代码新建了一个user类，该类继承自models.Model, 同时为该类是设置了两个属性姓名和电话。接下来就到了本节重点要概述的数据库操作过程。正如本文前面对Django进行介绍时提到的，CURD模式没有直接操作SQL语句的地方，都是对象化操作，就像调用函数一样简单，例如如ResModel，执行一次ResModel()就是新建一行记录，代码如下。

|  |
| --- |
| a=ResModel()  a.filename="文件名"  a.file="文件对象"  a.save() |

这样就完成了一行记录的添加，中间完全不需要写SQL语句。一个表的创建也无需SQL操作，在models.py里面写上需要建立的表，然后执行一次python manage.py migrate就可以创建自己需要的表。这之前需要运行一次python manage.py makemigrations。这样的操作可以看出在sqlite里面一个class就是一个表，因此对数据库的操作更像是面向对象的函数操作。下面就给出本课题的数据库操作代码models.py。数据库的操作需要提前在models.py中引入其他模块，因为数据库的操作方式是面向对象的，因此只有引入其他模块之后才能将要操作的数据对象化存进sqlite数据库。代码如下。

|  |
| --- |
| from django.db import models |

以下是对数据库的操作包括建表以及数据表操作过程, 在给这些数据对象化的过程中必然也要对这些对象进行属性定义和赋初值。以下是普通用户数据表的创建过程。

|  |
| --- |
| class UserModel(models.Model):  name = models.CharField(max\_length=50)  icon = models.ImageField(null=True, upload\_to="avaters")  user = models.ForeignKey(User, on\_delete=models.CASCADE)  resRoot = models.ForeignKey(ResModel,   tel = models.CharField(max\_length=50)  credit = models.FloatField(default=0)  address = models.CharField(max\_length=150) |

需要注意到每一个属性的赋值都是对应的view里的一个表单或是文本域，通过表单或文本域收集到的用户输入数据在经过对对象的属性赋值以后存储进入数据表中，同时每一个字段的存储也满足规定的格式。整型数据字段的输入也要满足限定的长度范围。其余表的创建过程十分相似，这里不再概述，以下是所有数据表的表结构图。如图4.1。

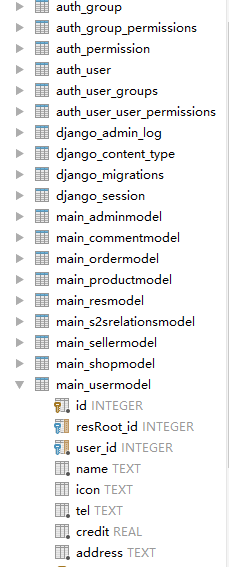


图4.1 数据表结构图

通过以上的数据库建立过程可以看到，在Django中数据库的操作更多是面向对象的编程过程，每一个数据表更像是一个类而并非通过标准的SQL语句进行数据库的操作。数据库中的字段则是通过对对象属性的定义进行设置和赋值的，这个开发人员降低了很大的难度，也帮助开发人员更加专注于自己所关心的业务逻辑方面。

## 4.2 用户登录模块的实现

用户要进入本系统进行操作，就需要通过登录模块的身份认证，登录模块本质就是一个用户身份比对验证的过程，这一节将对登录认证模块的实现进行概述。登录模块的实现过程是从用户信息数据表的建立开始的，用户数据表的建立过程前面在数据库创建小节中已经介绍这里不再重复。

参照前面分析得到的流程图，完成其具体过程，如下：

（1）从对话框中读取用户输入的用户名和密码。

（2）检查输入的用户名和密码是否为空，也就是检查数据的有效性。

|  |
| --- |
| {% if login\_errors%}#判断上次的登录请求是否有错误产生  {{ login\_errors.username }}  {{ login\_errors.password }}  <div class="am-form-group">  <input name="username" class="am-input" type="text" placeholder="用户名" value="{{ username}}" /></div>  <div class="am-form-group">  <input name="password" class="am-input" type="password" placeholder="用户密码" /></div>  <div class="am-form-group">  <label class="am-checkbox"><input id="auto-login" type="checkbox"/> 下次自动登陆</label></div>#本次不成功返回提示信息 |

（3）根据用户名从用户信息表中取得密码。

（4）判断从数据库中取得的密码和用户输入的密码相同，如果相同，则能进入系统，如果不同，则返回。

|  |
| --- |
| if UserModel.objects.filter(user=request.user).count(): return redirect("membercenter")#输入的账户存在，返回登陆成功。 auth\_logout(request)#注销 … ifUser.objects.filter(username=form.cleaned\_data["username"]).count():  #若输入账户无效进入注册页面或者用户直接选择进入注册页。form.errors["username"] = ["用户名已存在"]#用户名已经被注册 u.username = form.cleaned\_data["username"]#填写用户名 u.set\_password(form.cleaned\_data["password"])#填写密码 user.tel = str(form.cleaned\_data["tel"])#填写电话 return render(request, "register.html", context=context)#信息合法返回登录页面进行登陆。 |

程序将打开登录对话框，用户只有在通过身份认证后才能初始化主对话框。系统的登录界面如图4.1所示。



图4.1　登录模块主界面

## 4.3 文件管理模块的实现

个人用户文件管理模块包括文件管理、文件存储、积分系统三个小部分。其中文件管理部分可以实现文件的上传下载、文件夹创建、查看详情等。下面将对三个模块依次分解概述。

4.3.1 文件上传下载模块的实现

文件的上传首先需要在前端界面预留交互接口点击Button之后弹出文件选择窗口，在用户查找到自己要打印的文件之后，点击上传在文件上传成功之后就可以在view视图中看到已经成功上传的文件。如果文件上传不成功则返回，失败提示，并建议用户重新上传。文件上传的实现原理是，首先通过if re.count()函数预判要上传的文件进行判断，判断其类型，这里为文件、文件夹分别设置了标识符0和1。若是0则可以判断文件类型是的文件夹，若是1则为独立文件。文件存储函数的实现原理是给每一个要上传的文件赋值一个id号码[24]。文件上传前存储函数会首先预判其上传的上级目录，文件上传时函数根据已获取的目录id给新上传文件分派一个新的基于上级目录的id号码，将这个id文件存于文件表的res\_id字段，同时将文件以TEXT数据类型存储进res数据表中。

|  |
| --- |
| def uploadRes(request):#定义文件上传的模块  ...  if res.count():#判断文件类型 …  children = [] if res.children=="" else res.children.split(",")#判断文件夹还是文件 … file.save()#将文件存储 res.children = ",".join(children)#将文件传入文件夹中 |

文件下载过程同样如此，不同的是下载时用户除了可以通过文件夹路径进行多级查找之外，还可以通过搜索框进行文件搜索，依次快速定位要打印的文件。这样能省去不少查找时间。文件搜索功能实现原理相对简单。

|  |
| --- |
| def downloadRes(request, resId):#定义下载模块 … return redirect(file.file.url) |

4.3.2 文件夹创建和文件搜索模块的实现

文件夹创建过程略显复杂，文件夹创建的过程是一个主观性通过系统动态代码生成一个新的文件的过程。这个文件在系统中被定义为一个对象，每生成一个文件夹，则生成一个新的文件夹对象。接下来在对文件夹对象进行简单的概述，首先每创建一个文件夹对象就相当于生成了一个根文件目录，这有一点比较类似于生成linux系统下的. /文件目录。凡是存在该目录下的其他文件，包括文件夹。都成为了该文件夹的children。这样如果是多层文件夹嵌套。在用户点击了文件夹之后，view.py则会返回该文件夹的children文件，如果其children文件依然是文件夹则view.py会不断循环返回一个文件夹对象。直到其children文件是一个文件，这时继续点击则会返回该文件的详情页面。以下是文件夹的创建过程。以下是文件夹创建过程部分的部分实现程序。

|  |
| --- |
| …  children = [] if res.children=="" else res.children.split(",")#判断文件是否有下级并取出这些下级  …  "icon": "am-icon-file" if resC.type else "am-icon-folder",#对象应该显示的图标 "name": resC.filename,#设定对象应该显示的名称  …  def browseFiles(request):#以下是文件搜索部分的部分实现程序。  …  return HttpResponse("文件路径错误，请刷新重试！") |

如图4.2是文件管理器的部分截图，可以看出本在线文件管理系统基本完全具备桌面系统资源管理器的功能。涵盖了文件增删改查十分全面。唯一不同的是本资源管理器不可使用右键唤出详细操作菜单，而是通过文件右上角的操作按钮展开详细操作。



图4.2 文件管理器界面图

4.3.3 文件进入打印流程及查看详情模块的实现

通过上面的文件检索进入具体的自己要查找的文件之后通过单击文件或者点击文件右上角的操作按钮都可以进入文件打印和文件详情查看的界面。之所以将文件打印前设置功能与文件打印入口进行分离，考虑到打印前设置是一个订单生成的过成。这个过程是一个对打印文件进行贴标签的过程，因此按照软件工程的高内聚低耦合的原则，本课题将这一模块划分开来进行单独处理。以下是文件进入打印流程及查看详情模块的实现程序。

|  |
| --- |
| … context["res"] = { "filename": res.filename,#获取文件名称 "size": res.file.size,#获取文件大小 "date": res.date,#获取文件上传日期 "printCnt": res.printCnt,#获取文件打印次数 "shareCnt": res.shareCnt,#获取文件分享次数  … |

获取文件大小及文件上传日期的函数的原理都是在文件上传时就一次性获取了浏览器判断到的瞬时值，进而将获取到的文件大小、上传时间进行存储。而文件下载、打印次数统计则需要函数动态的在每一次文件被分享、下载后进行一次累计，并实时将累计结果更新至view中显示在用户界面上。如图4.3是文件详情查看的部分界面图。



图4.3 文件详情功能实现图

## 4.4 打印前设置模块的实现

打印前文件设置就是一个对要打印的文件贴标签的过程，具体打印前设的程序设计原理就是，为每一个要打印的文件生成自己的下载url以这个url为基础设置过程每一次为文件附加一项属性，则为该对象属性附一次值。以下是打印前设置功能实现的程序。

|  |
| --- |
| def printRes(request, resId):#定义打印函数首先获取文件id user = UserModel.objects.get(user=request.user)#获取下单用户的id … "credit": user.credit,#从用户积分池中扣除积分 "shops": ShopModel.objects.all(),#将该订单加入店铺模型相应的部分 …  "name": res.filename,}#获取文件名称 return render(request, "udisk/print.html", context=context)#成功后返回网盘页面 |

如图4.4是这个打印设置流程界面截图。





图4.4 打印设置流程界面

## 4.5 商家后台模块的实现

商家后台模块要分为四个小模块，分别为订单管理模块、店员管理模块、积分管理模块、商品发布编辑模块。下面将直接对每一个小模块进行实现概述。

4.5.1订单管理模块

订单管理模块是文件打印前设置的末端模块。这一模块是对所有个人用户发至商家的订单的一个总体汇总。商家通过订单管理模块可以进行订单状态的确认。订单状态的改变原理是通过为每一个订单打上一个标签状态变量status，并令该标签状态变量随着用户的提交submit或者商家的手动改变click而变化。同时在view层为了让用户清楚的知道该订单状态已经变化，在程序上通过三个可以随着不同的status值发生变化的<ol>标签进行显示。例如，在订单处于未支付状态时order.status == 0，此时令<ol>标签的颜色透明、订单处于已支付待商家确认的状态时order.status == 1，此时令<ol>标签的颜色为红色等。具体的程序实现如下。

|  |
| --- |
| … {% if order.status == 0 %}等待付款 {% elif order.status == 1 %}等待商家确认 {% elif order.status == 2 %}商家正在打印  {% elif order.status == 3 %}文档等待领取 {% elif order.status == 4 %}订单已完成 {% elif order.status == 401 %}商家无法打开您的文件 {% elif order.status == 402 %}订单已关闭 {% elif order.status == 403 %}订单正在退款  {% elif order.status == 405 %}订单已退款 |

订单模块功能实现如下图4.5所示。进度条的长度不同颜色也不同，就表示不同的订单状态。

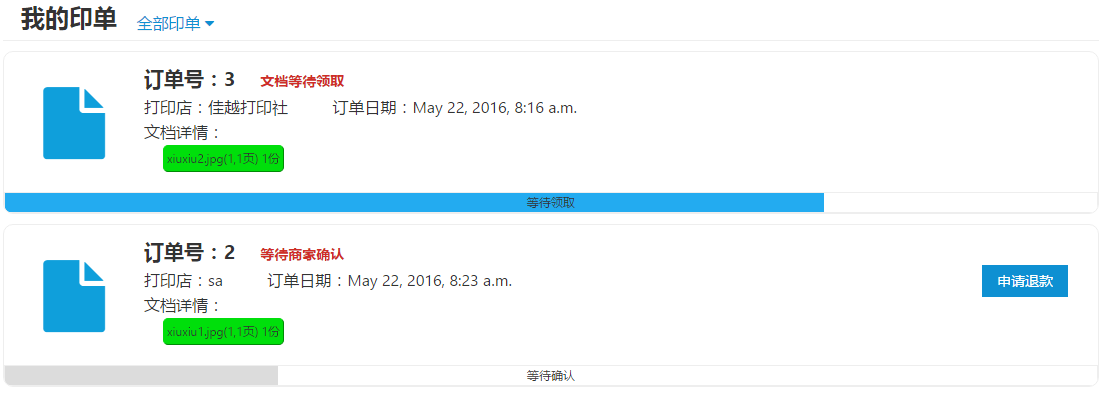


图4.5 订单模块图

4.5.2店员管理模块

店员模块的设计原则是完全模拟现实中打印店的员工设置。例如，店员分为店长和普通员工，店长在整个店铺日常管理中权限最高，店员次之。具体的程序实现原理是，在店员账号注册时注册项里会包括店铺信息的填写。因此在一个店员账号注册成功的同时一个店铺也注册成功了，这时候在用户下单过程中就会在下拉选项中看到新注册的店铺。但是一个店铺注册只能在第一个注册该店铺的店员账号生成时而生成，而这第一个注册店员也默认为店长。之后的普通店员账号的增加可以通过店铺后台页面的店员添加按钮进行新店员账号的添加。此时新店员的添加需要注意的是在除了填写用户、密码、电话等基本信息的同时还可以选择该账号类型为普通店员还是店长。这时如果将新账号设置为店长此时的新账号就也具有了管理店铺的最高权限级别。具体程序实现如下。

|  |
| --- |
| … <label><input type="radio" name="job" value="0"  {% if not recruit.job %}checked{% endif %}> 员工</label> <label><input type="radio" name="job" value="1"  {% if recruit.job %}checked{% endif %}> 店主</label> |

下面是店员管理模块的部分实现界面。

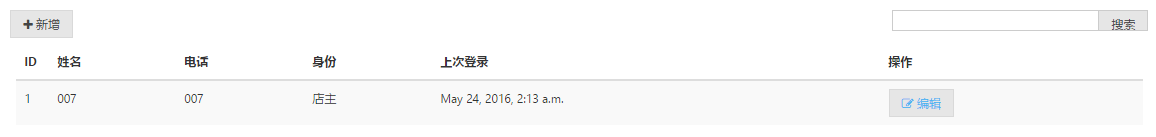


图4.6 店员账号管理图

4.5.3积分管理模块

积分管理模块是内建的虚拟交易货币，首先实现时定义两个字段分别为商家端的集分变量，再定义一个用户端的积分变量，两个变量都可以在接受支付宝充值之后经过对应的积分换算公式而增加。用户端的积分变量会随着每一次下单扣除的相依的积分额度产生减少。商家端则会在提现之后发生减少。每一次下单随着用户端的额度减少对应的数值之外，商家端相应的会增加对应的额度。具体实现部分的代码如下。

|  |
| --- |
| #这里在view中对积分的提现比例做一个说明。  <span style="color: #bbb; margin-bottom: 5px;">（100分 = 1元）</span>  #以下是商家端决定订单order增减的代码算法。  … seller = SellerModel.objects.get(user=request.user) order = OrderModel.objects.get(id=orderId)  …  elif orderAction == "taken": order.status = 4 order.shop.credit = order.shop.credit + order.credit order.shop.save() order.save() return redirect("history-orders")#所有历史订单 |

部分实现界面图如图4.7积分充值，提现部分截图。



图4.7 积分充值提现功能图

## 4.6商品编辑发布模块的实现

商品编辑发布模块用以编辑商品信息，首先需要一个在线文档编辑工具对要发布的商品详情进行编辑，本课题开发中用到了第三方的在线编辑框架ueditor。使用该框架可以满足桌面编辑器的多数功能，如文字大小调整、颜色调整、对齐方式、插入图片等等。按照编辑模块的api要求将该编辑工具引入系统。引入后，系统需要能保存编辑后的文本内容并将其发布之商城前端页面进行同步显示[23]。ueditor引入后的系统效果图如图4.8 所示。

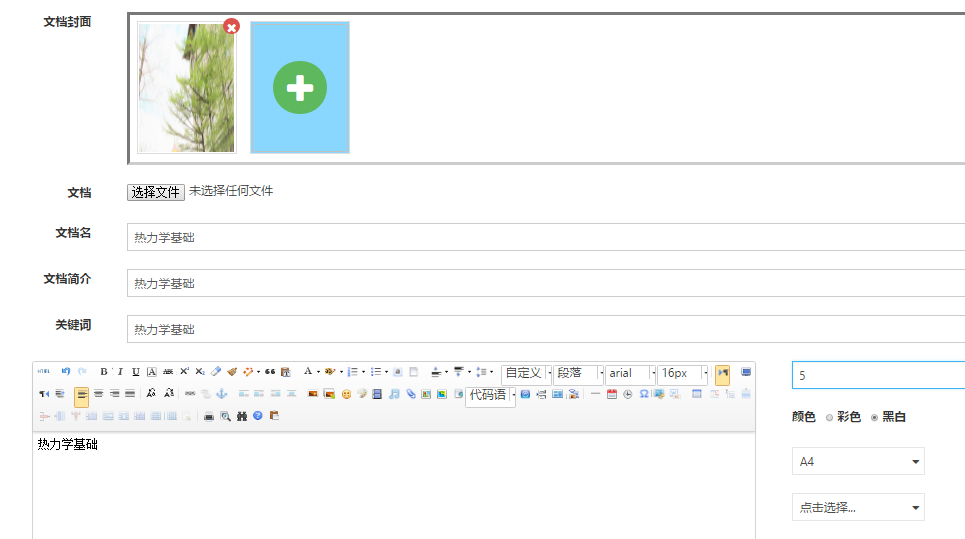


图4.8 ueditor引入效果图

## 4.7资源交流社区模块

资源交流社区用到的开源的discuz系统，但是考虑到本社区的定制性要求，因此本课题在社区模块实现过程中个性化定制了专用模板。下面这一小节将对这一模板页面的开发设计和定制部署过程进行概述。在开发discuz模板之前首先需要了解discuz的模板文件的文件结构。以下是discuz!的几个模板文件。

（1）header.html页面头部

（2）discus.html论坛首页

（3）footer.html页面底部

（4）viewthread查看帖子

首先，现在将要开发的这模板命名为yxpi，并在templates目录下建立一个yxpi目录。在yxpi目录下再新建一个images文件夹用于存放这套模板的图片，把切好的所有图片保存到这个文件夹中。templates/test文件夹用于存放这套模板的htm文件，在这个目录中新建三个文件header.htm、discuz.htm、footer.htm，打开开发完成的主页html文件，将要作为页面头部的代码复制到header.htm文件中，相应的将作为底部的那一段代码复制到footer.htm文件中，其余的代码复制粘贴到discuz.htm文件中。最后在discuz.htm文件的最顶部和最底部分别加上：{subtemplate header} 和{subtemplate footer}这两个标签，这是是discuz的规范的标签语言，意思是将header.htm和footer.htm文件包含进来，这样就构成了一个完整的主页面了。

部署完毕html文件之后就可以将开发完成的CSS文件部署在模板目录下（templates/test），将其重命名为“css\_append.htm”（注意：改后的文件名是css\_append，后缀名是htm）。接着打开header.htm文件，找到需要引入CSS文件的语句，于是本系统中的<style type="text/css">@import url(css/style.css);</style>将这个替换为一下代码。

|  |
| --- |
| $rsshead $extrahead {subtemplate css\_script} |

这样就引入了CSS文件。进入后台更新一下缓存，尽可以看到与静态相同的效果图了以上是discuz的前端框架调用原理，在了解了这些之后就可以着手开发本系统要用的模板文件。根据上一章的功能需求分析，及系统流程分析，可以讲在线资源分享社区分解为资源及评论发帖区模块，分组讨论区模块。discuz框架具有成熟的后台交互逻辑以及成熟的数据库文件，因此在使用该系统时需要大量投入精力进行研究的是系统模块设计及前端交互界面的开发。前端界面开发完成只需按照上面的流程将文件引入discuz就可以得到满足系统需求的论坛系统了。

## 4.8 本章小结

本章是系统设计的开发实现。开始先介绍了数据库的组建过程。进而介绍了系统用户登录模块、文件管理模块、打印前设置模块、商家后台模块、在线商城模块、最后介绍了资源交流论坛模块。

# 结 论

本系统作为在线的O2O文件打印系统很好的结合了当前社会的发展需求，准确的挖掘当前许多学生、教师、办公族的便捷打印需求。O2O在线打印系统能够提高打印店铺的工作效率，令商家更好的组织日常打印任务。同时为有需求的个人群体节约了大量的学习工作时间。这既符合当前O2O模式发展的方式，又适应现代人日常消费习惯。在本课题的设计实现以数据库和后台逻辑关系为主，通过前期数据库设计过程很快的勾画出了整个系统的各个角色的用例关系。开发实现的过程可以概括为，数据库的设计实现，Django框架的部署，各模块的划分与开发。

通过开发本系统，能够实现用户打印文档的在线存储，实现打印店铺规范化合理打印日常打印任务。同时实现店铺日常各级员工分级管理店铺日常活动，通过在线商城的建立可以实现商家将无打印任务时时闲置的打印资源最大化利用。同时普通个人用户又可以通过在线资源交流社区进行资源和信息交流，实现信息共享。

本课题的开发过程，遵循客观的软件工程开发流程。从系统需求分析、数据库模型设计、程序的开发等这些完整的过程，都严格遵守开发规范，确保系统软件质量优良。受限于时间和学习资历，系统仍有许多方面有待更好的改良。如，在选择附近的打印店铺过程中如果能实现地图自动定位周边店铺将会在用户体验上有更佳的改善。对于这些也将是今后学习的方向和研究的动力所在。

# 参考文献

[1] 张波. O2O:移动互联网时代的商业革命[M].北京:机械工业出版社, 2013:21-31.

[2] 布朗. Python技术参考大全[M].北京:清华大学出版社, 2002:34-37.

[3] 卢茨. Python编程[M]. 中国电力出版社, 2015:45-56.

[4] Adams P D, Afonine P V, Bunkóczi G, et al. PHENIX: a comprehensive Python‐based system for macromolecular structure solution[M]// International Tables for Crystallography. John Wiley & Sons, Ltd, 2012: 213–221.

[5] TOBY DONALDSON(加). PYTHON编程入门(第3版)[M].北京:人民邮电出版社, 2013.55-60

[6] Johansson J R, Nation P D, Nori F. QuTiP 2: A Python framework for the dynamics of open quantum systems [J]. Computer Physics Communications, 2013, 184(4):1234-1240.

[7] Lutz M. Learning Python 4th Edition[J]. Oreilly Media, 2009.

[8] 刘祺彬.基于开源技术的MVC框架的构建[J].信息技术,2013(8):125-129.

[9] XU Tong-de, LI Fa-chun. Design and implementation of project management system Based on the three layer architecture and MVC pattern[J]. Journal of Anhui Vocational College of Electronics & Information Technology, 2016.

[10] 曹骏.基于Django的数据收集系统的设计与实现[D].南京:南京大学,2013.

[11] JEFF FORCIER(美). DJANGO WEB开发指南[M].北京:机械工业出版社, 2009.

[12] Bennett J. Practical Django Projects, Second Edition[M]. Apress, 2009.

[13] Holovaty A, Kaplanmoss J. The Definitive Guide to Django: Web Development Done Right[J]. 2009.

[14] Nelson J. Building a Library App Portfolio with Redis and Django[J]. Code4lib Journal, 2013(19).

[15] K. Charchuła, Schuler G A, Spiesberger H. Combined QED and QCD radiative effects in deep inelastic lepton-proton scattering: the Monte Carlo generator DJANGO6[J]. Comput.phys.commun, 1994, 81(3):381-402.

[16] 丁月光.PHP+MySQL动态网站开发[M].北京:清华大学出版社,2008:23-34.

[17] 王石,杨英娜.精通PHP+MYSQL应用开发(附光盘)[M].北京:人民邮电出版社, 2007:23-25.

[18] Williams G. HTML5[M]. Apress, 2012.

[19] Flanagan D P. JavaScript: The Definitive Guide[M]. O'Reilly & Associates, Inc. 1998.

[20] (美)苏灵 (Suehring, S.). JavaScript从入门到精通[M].北京:清华大学出版社, 2014:66-67.

[21] Li N. By comparison to the ORM database modeling approach with the traditional approach[J]. Journal of Shaoyang University, 2007.

[22] Volder K D. JQuery: A Generic Code Browser with a Declarative Configuration Language[J]. Lecture Notes in Computer Science, 2006, 3819/2006:88-102.

[23] Perich F. Responsive design with Twitter Bootstrap: a tutrial and demonstration[J]. 2014.

[24] Paulson, Linda Dailey. Building Rich Web Applications with Ajax. Industry Trends[J]. Computer, 2014, 38(10):14-17.

# 攻读学士学位期间发表的论文和取得的科研成果

# **致　　谢**

四年的本科生活即将结束，尽管大学生活学业繁重，但是收获的确是宝贵的学习经验和无价的知识。这四年的生活里，我得到了来自家人、朋友、老师们的关怀、鼓励和教导。在这里我谨代表个人对你们真挚的感谢！

首先要感谢在这四年里给予我教育和培养的大学，它不仅教育我受用终身的专业知识，也在我的品行和处事方式上给予我莫大的帮助。

本论文在王红滨老师的精心指导下完成。无论是从最初的选题、立题到课题的设计实现以及最终的论文修改完善，王老师都倾注了极大的精力。这期间的每一个阶段王老师都能及时有效的给我提出宝贵而有效的改进意见。同时在我每一次遇到难题是王老师总是能耐心备至的为我一一解答。王老师对我的教导来自专业技能、科学知识也来自于他优秀的治学风格。这期间王老师让我感受到了他深刻的治学经验，开阔的专业视角，以及扎实渊博的学识。毕设工作辛苦且周期较长，对于王老师在我生活和学习上的帮助我感到深深的感激之情。

也感谢在此期间在生活上给予我帮助的室友。和我一起商量问题的实验室同学徐衍鹏、张伟、齐志鑫，在背后默默支持我的蒋建同学。本论文也得到了研究生师兄的指导，在此表示感谢。在即将毕业之际，还要感谢其他和我一起学习生活了四年的同班同学们，感谢他们在学习上对我的帮助，在生活上对我的包容和照顾。最后感谢我的父母，他们的关怀和鼓励是我最大的精神支柱也是我前进的动力，他们对我的希望，激励我奋发前行。

从选定课题到论文的顺利完成，有很多博学的师长、友好的同学、温暖的朋友以及爱我的家人给了我很大的支持与帮助，在这里请接受我诚挚的谢意！