摘要

随着经济实力的飞速发展，人们的物质需求也在随经济发展不断提高。从过去的吃饱，到现在人们渴望吃的更好，更健康，更放心。最健康的方式某过于自己选购食材，自己烹煮加工。但忙碌的工作生活中难以抽出时间来去菜市场选购优质的食材。这时时下流行的新零售的模式下的生鲜电商就如雨后春笋般开始蓬勃发展。

在被称为新零售元年的 2017 年，各种新零售创新业态迎来集中爆发。到 2018 年更是发展的愈加汹涌。线下永辉超级物种，天虹 space，新华都海物会，步步高鲜食演义，百联 RISO，大润发优鲜，世纪联华鲸选，物美多点等相继出炉。线上美团掌鱼生鲜，苏宁苏鲜生也纷纷亮相，加上此前开业的阿里盒马鲜生，以及今年年初开业的京东 7FRESH，短短两年时间，主要新零售超市已开出 100 家（数据截止 2018 年 2 月 2 日）。

在这些生鲜电商中，有传统零售业如大润发，世纪联华等的参与，也有互联网巨头阿里，京东等的参与，进一步说明人们这一块的巨大需求。

本文介绍了生鲜电商的开发环境，介绍了基于 iOS 平台 App 的架构，介绍了 App 的设计与开发以及各个子模块的功能。也有界面的设计与功能。

关键词：iOS 平台，生鲜电商，移动开发

ABSTRACT

With the rapid development of economic strength, people's material needs are also constantly improving with economic development. From the past fullness, people are now eager to eat better, healthier and more at ease. One of the healthiest ways is to use more of your own ingredients and cook it yourself. However, it is difficult to find time in the busy working life to go to the food market to purchase quality ingredients. At this time, fresh e-commerce under the popular New Retailing model is booming.

In 2017, known as the first year of new retail, various New Retailing innovations ushered in a concentrated outbreak. By 2018, development is even more surging. Under the line, Yonghui Super species, Tianhong Space, Xinhuadu Sealife Club, Backgammon Fresh Foods, Brilliance RISO, RT-Mart Fresh, Century Lianhua Whale Selection, Wumart Multi-point, etc. Fresh fish of the U.S. group were bred freshly, and Suning Sue fresh appeared. With the opening of the Ali He Xiangma freshman and Jingdong 7FRESH, which opened earlier this year, 100 major new retail supermarkets have been opened in just two years(Data as of February 2, 2018).

Among these fresh e-commerce providers, participation from traditional retailers such as RT-Mart and Century Lianhua, as well as the participation of Internet giants Ali and Jingdong, further illustrate the huge demand of people.

This paper introduced the development environment of the fresh e-commerce business, introduced the architecture of the iOS-based App, and introduced the design and development of the App and the functions of each sub-module. There are also interface design and functions.

Keywords: iOS platform, Fresh electricity supplier，Mobile development

目录

第 1 章 引言

1.1 背景

2016 年 11 月 11 日,国务院办公厅印发《关于推动实体零售创新转型的意见》（国办发〔2016〕78 号)。《意见》在促进线上线下融合的问题上强调：“建立适应融合发展的标准规范、竞争规则，引导实体零售企业逐步提高信息化水平，将线下物流、服务、体验等优势与线上商流、资金流、信息流融合，拓展智能化、网络化的全渠道布局。”

在当前电商发展现状中，由于目前传统线上零售遭遇天花板，移动支付等新技术开拓线下场景智能终端的普及，以及新中产阶级的崛起等种种原因，促进了将线上线下和物流结合在一起的新零售的产生。在 2016 年 10 月的阿里云栖大会上，阿里巴巴董事长马云在演讲中第一次提出了新零售，“未来的十年、二十年，没有电子商务这一说，只有新零售。借此我们想通过自己来设计一个电商零售 App，来感受如何从传统电商中走向新零售模式下的电商发展模式。

本平台也是在零售快速变革发展的大环境下，为用户提供更加优质的购物体验，达到零售的新高度。

1.2 研究内容

本 App 的设计的初衷是制作一个拥有全新体验以及符合当下新零售模式的生鲜购物 App。本 App 是基于 iOS 平台[1]，当下由于移动支付等新技术开拓线下场景智能终端的普及，绝大部分的购物行为以及支付行为都发生在移动设备上，其实移动设备中 iOS 又占据半壁江山，所以本次研究选择了在 iOS 平台开发。相较于传统电商 App，例如淘宝这类全平台的购物平台。

本次研究的内容将购物的范围限制在了人们每天接触更多的生鲜类产品为主。同时这么做的也更符合新零售的定义，线上线下和物流结合在一起才会产生新零售。而不是传统零售业一样，将物流交付给以顺丰为首的物流公司。本次开发中希望通过界面设计上的优化，来提高用户的使用体验，让用户更加适应此类新的零售模式[2]。

第 2 章 开发环境及技术介绍

2.1 开发环境

开发语言：Swift + JavaScript

开发工具：Xcode + Visual Studio Code

数据库：Mysql5.5 版本

服务器：Node.Js +阿里 Egg

调试工具：Reveal + Chrome 浏览器+Postman

2.2 开发技术介绍

2.2.1 Swift 语言

Swift 是一种支持多编程范式和编译式的编程语言，是用来撰写 macOS/OS X、iOS、watchOS 和 tvOS 的语言之一。 2014 年，其在苹果开发者年会（WWDC）发布。设计 Swift 时，苹果公司有意让 Swift 与 Objective-C 共存在苹果公司的操作系统上[3]。

苹果宣称 Swift 的特点是：快速、现代、安全、互动，而且明显优于 Objective-C 语言。Swift 以 LLVM 编译，可以使用现有的 Cocoa 和 Cocoa Touch 框架。Xcode Playgrounds 功能是 Swift 为苹果开发工具带来的最大创新，该功能提供强大的互动效果，能让 Swift 源代码在撰写过程中能即时显示出其运行结果。拉特纳本人强调，Playgrounds 很大程度是受到布雷特・维克多理念的启发。

Swift 语言在经过 4 年的发展，现在已经有越来越多的企业开始将公司的 App 从 Objective-C 语言向 Swift 语言转换，同时因为 Swift 语言尚属比较年轻的语言，发展过程中常常会带来一些令人激动的新特性。

2.2.2 JavaScipt 语言

JavaScript，一种高级编程语言，通过解释执行，是一门动态类型，面向对象（基于原型）的解释型语言。它已经由 ECMA（欧洲电脑制造商协会）通过 ECMAScript 实现语言的标准化。它被世界上的绝大多数网站所使用，也被世界主流浏览器（Chrome、IE、Firefox、Safari、Opera）支持。JavaScript 是一门基于原型、函数先行的语言，是一门多范式的语言，它支持面向对象编程，命令式编程，以及函数式编程。它提供语法来操控文本、数组、日期以及正则表达式等，不支持 I/O，比如网络、存储和图形等，但这些都可以由它的宿主环境提供支持[4]。

由于当前大前端的蓬勃发展，现在 JavaScript 语言也可以用于来编写服务器代码了。

2.2.3 Node.Js

Node.js 是一个能够在服务器端运行 JavaScript 的开放源代码、跨平台 JavaScript 运行环境。Node.js 由 Node.js 基金会持有和维护，并与 Linux 基金会有合作关系。Node.js 采用 Google 开发的 V8 运行代码，使用事件驱动、非阻塞和异步输入输出模型等技术来提高性能，可优化应用程序的传输量和规模。这些技术通常用于数据密集的事实应用程序[5]。

Node.js 大部分基本模块都用 JavaScript 语言编写。在 Node.js 出现之前，JavaScript 通常作为客户端程序设计语言使用，以 JavaScript 写出的程序常在用户的浏览器上运行。Node.js 的出现使 JavaScript 也能用于服务器端编程。Node.js 含有一系列内置模块，使得程序可以脱离 Apache HTTP Server 或 IIS，作为独立服务器运行。

目前，Node.js 已被 IBM、Microsoft、Yahoo!、Walmart、Groupon、SAP、LinkedIn、Rakuten、PayPal、Voxer 和 GoDaddy 等企业采用。

2.2.4 阿里 Egg 框架

Egg 是阿里基于 Node.js 上开发一个企业级后台框架。Egg 拥有自由的插件扩展机制，旨在一个插件只做一件事。Egg 通过框架聚合这些插件，并根据自己的业务场景定制配置，这样应用的开发成本会有所降低。

同时 Egg 奉行约定优于配置，按照一套统一的约定进行应用开发，团队内部采用这种方式可以减少开发人员的学习成本，开发人员不再是“钉子”，可以流动起来。因此 Egg 框架具有以下特性：提供基于 Egg 定制上层框架的能力，高度可扩展的插件机制，内置多进程管理，基于 Koa 开发，性能优异，框架稳定，测试覆盖率搞，渐进式开发等特性。

2.3 开发工具介绍

2.3.1 Xcode

Xcode 是苹果公司向开发人员提供的集成开发环境，用于开发 macOS、iOS、WatchOS 和 tvOS 的应用程序。Xcode 前身是继承自 NeXT 的 Project Builder。

The Xcode suite 包含有 GNU Compiler Collection 自由软件（GCC、apple-darwin9-gcc-4.0.1 以及 apple-darwin9-gcc-4.2.1，默认的是后者），并支持 C 语言、C++、Fortran、Objective-C、Objective-C++、Java、AppleScript、Python、Ruby 和 Swift，还提供 Cocoa、Carbon 以及 Java 等编程模式。协力厂商更提供了 GNU Pascal，Free Pa**scal, Ada, C#, Perl, Haskell** 和 D 语言。Xcode 包使用 GDB 作为其后台调试工具[7]。

2.3.2 Visual Studio Code

Visual Studio Code（简称 VS Code）是一个由微软开发的，同时支持 Windows、Linux 和 macOS 操作系统且开放源代码的文本编辑器。它支持调试，并内置了 Git 版本控制功能，同时也具有开发环境功能，例如代码补全（类似于 IntelliSense）、代码片段、代码重构等。该编辑器支持用户自定义配置，例如改变主题颜色、键盘快捷方式、编辑器属性和其他参数，还支持扩展程序并在编辑器中内置了扩展程序管理的功能。

第 3 章 系统可行性和需求分析

3.1 系统可行性分析

3.1.1 技术可行性

本 App 采用 Swift 作为 iOS 客户端的主要开发语言，以 JavaScript 为后台开发的主要语言，附以 Egg 作为后台的 Node.Js 框架。Mysql 作为数据库。到目前为止，在利用 Swift 开发 iOS 项目的体验已经完全可以与 Objective-C 相媲美了，也具备一定的成熟性了。一些比较主流的 iOS 框架现在都已经有 Swift 版本，且有更多年轻，更具活力的项目开始使用 Swift 作为主要的开发语言，因此 Swift 在基础框架上已经完全可以胜任开发要求了。

同时 Egg 作为后端框架的也是比较简单可靠的，Egg 基于 Koa 发展，Api 方面较为稳定，且更为通用。且 Egg 在对于数据 Mysql 方面上的支持更可以说是达到了官方支持的程度，对于企业级项目的开发完全没有人可以问题。利用 Egg 的 Mysql 插件可以非常简单实现数据库操作。

3.1.2 经济可行性

本 App 在开发过程中使用到语言以及框架均属于开源资源，所用的开发工具也都是免费使用的。用户使用本 App 所需要的设备只是一台搭载 iOS 系统的普通设备。

3.1.3 易操作性

本 App 在界面设计上以苹果官方的人机交互指南为设计的基准，符合用户对于 iOS 系统中 App 的操作逻辑。用户使用本 App 会非常流畅，极易上手，不存在入门门槛。

3.1.4 法律可行性

本 App 不存在任何违法行为，也不会成为任何违法的工具。

3.2 系统需求分析

新零售下的生鲜购物 App 旨在通过简洁大方的界面设计，以及优质的交互设计来为用户提供过去传统电商平台难以提供的的优质的，专业的，更加细分的服务。用户可以根据分类快速找到自己所需要的商品，也可以通过关键词的搜索快速定位商品。同时也可以从首页浏览到推荐的商品，从中进行挑选。在浏览商品的详情页后可以对商品进行收藏，或加入购物车等操作。用户在确认订单支付后，可以比较快捷的对订单进行管理。因此在设计时应该满足以下几个目标：

（1）界面设计美观，UI 交互逻辑清晰，信息查看的便捷，快速。数据存储安全，可靠。

（2） 在首页展示当时推荐的商品内容。

（3） 为用户提供一个分类选择的商品以及搜索商品的功能。

（4） 为用户提供一个购物车以及收藏商品的功能。

（5） 为用户提供一个可以查看并操作订单的功能。

（6） 对于用户的输入进行必要的检查，减少一些错误的发生。

（7）保证 App 运行稳定可靠。

3.3 功能需求分析

3.3.1 登录注册管理

为区分用户以及方便数据加密，每个用户都必须用一个账户，才可以使用本 App。注册方面，主要通过手机号，验证码，以及密码来完成注册的操作。登录时则通过手机号与密码进行登录。同时修改密码的操作也是通过验证手机验证码来实现对用户的鉴别。

登录注册 ER 图（如图 3-1 所示）：

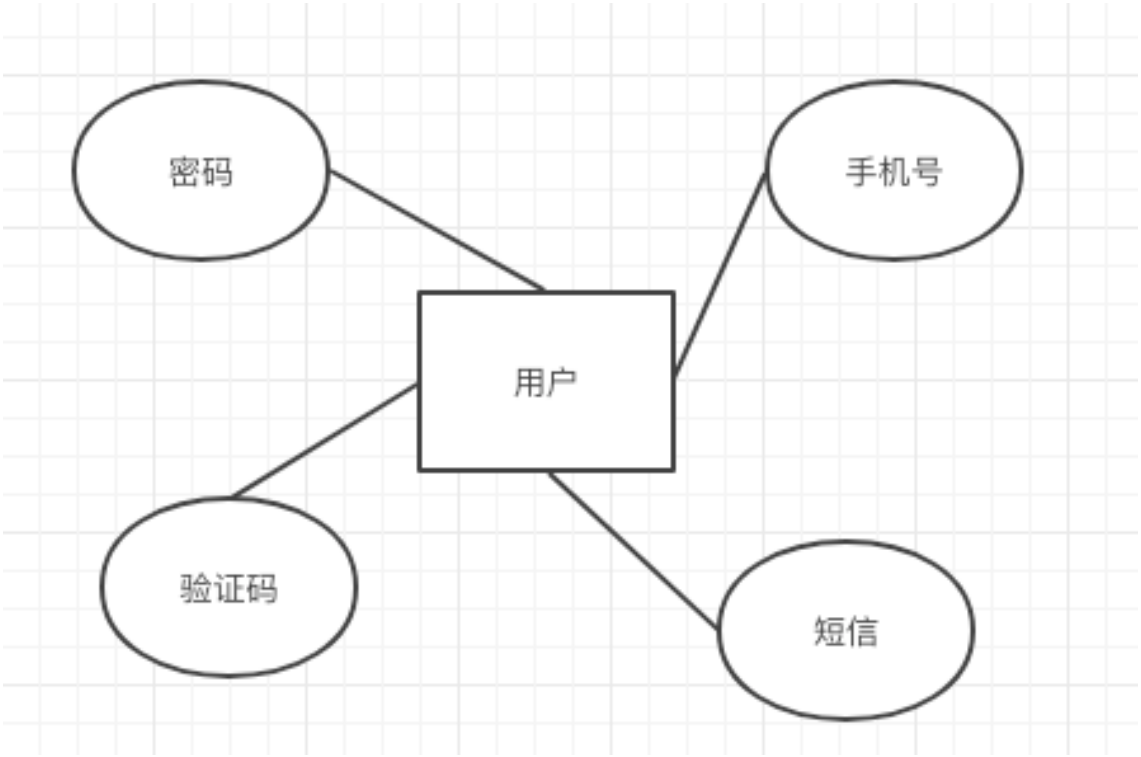

图 3-1 登录注册管理 ER 图

3.3.2 商品展示管理

本电商购物平台的对商品的展示，用户可以在首页浏览当前正在被推荐的商品，可以进入分类页面，根据商品的分类对商品进行定位，快速寻找某一类的商品。通过关键字进行搜索商品。

商品展示管理 ER 图（如图 3-2 所示）：

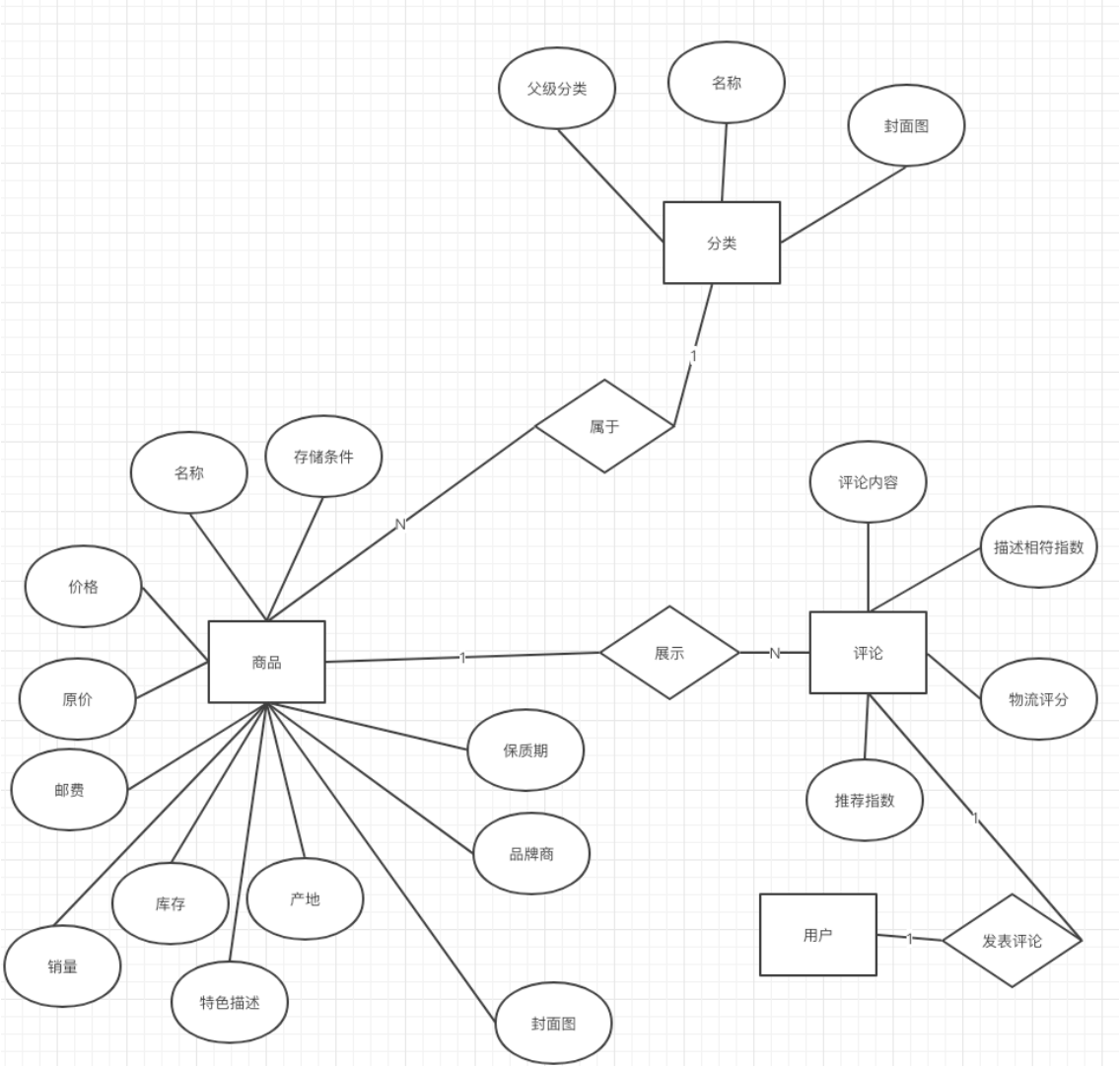

图 3-2 商品展示管理 ER 图

3.3.3 商品收藏管理

用户可以对商品进行收藏或取消收藏。同时可以在收藏列表中查看所有的收藏的商品，以及对所收藏的商品进行取消收藏操作，也可以从收藏列表进入商品详情。

商品收藏管理 ER 图（如图 3-3 所示）：

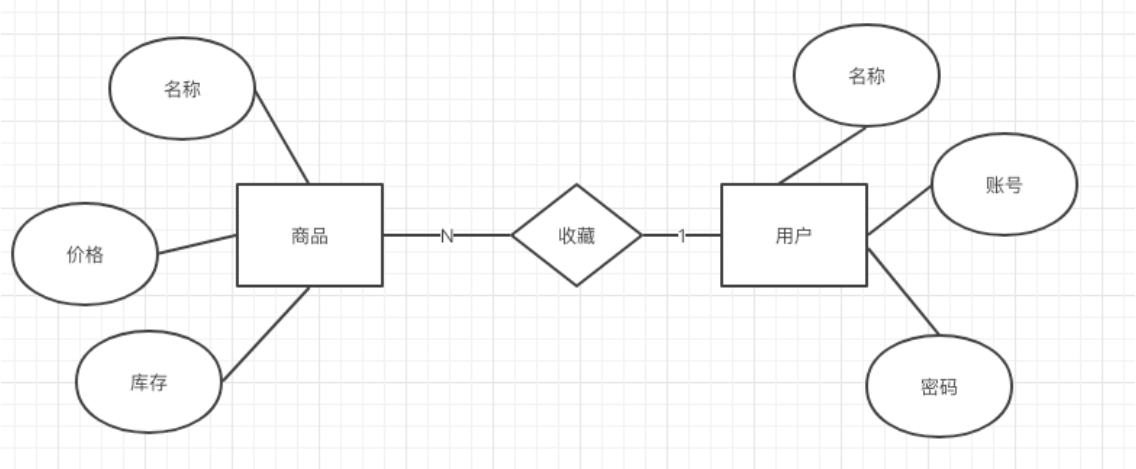

图 3-3 商品收藏管理 ER 图

3.3.4 购物车管理

用户可以在商品的详情页中将商品加入到购物车，在购物车页面中可以查看所加入到购物车的商品。

购物车管理 ER 图（如图 3-4 所示）：

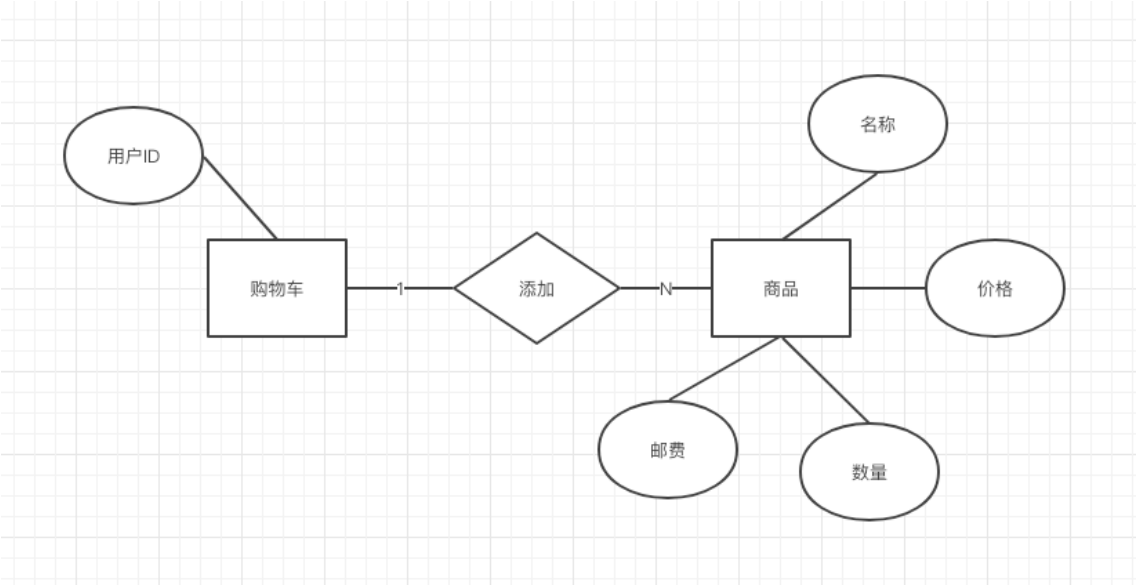

图 3-4 购物车管理 ER 图

3.3.5 订单管理

用户在商品详情页或是购物车页选择好商品后可以创建订单，并支付订单。在个人中心可以对订单进行支付订单，确认收货，发起退款订单的操作。

订单管理 ER 图（如图 3-5 所示）：

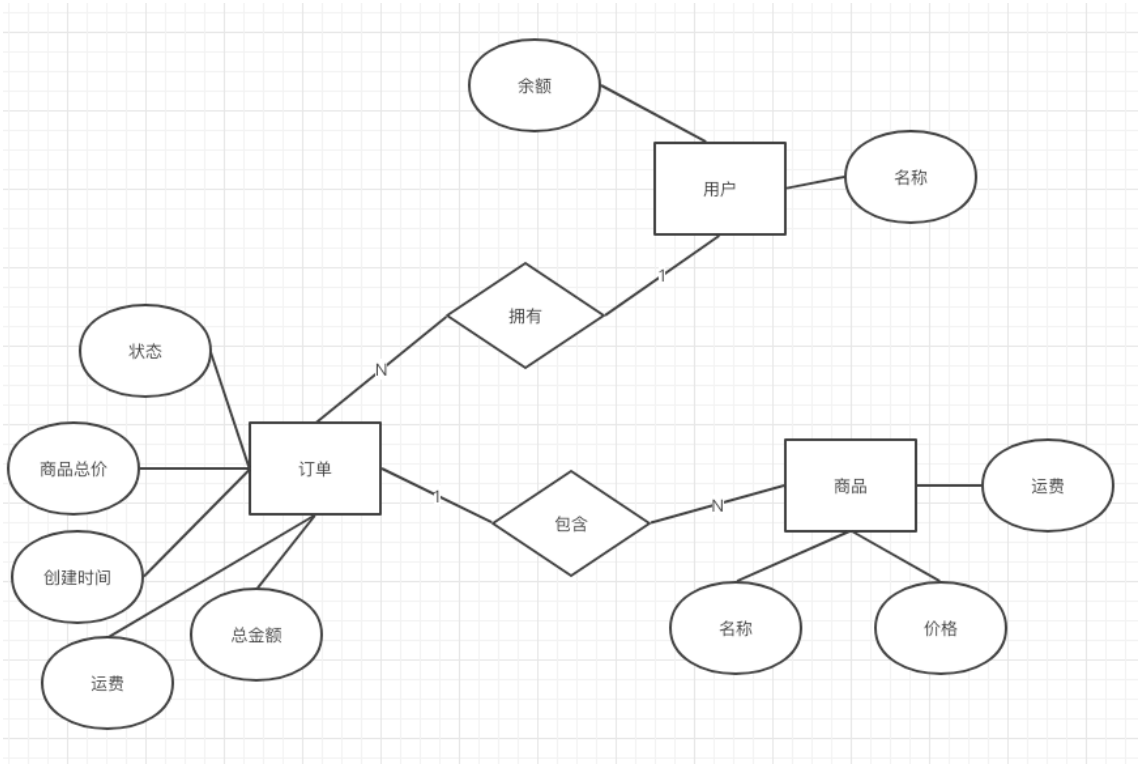

图 3-5 订单管理 ER 图

3.3.6 商品评论管理

用户在完成一个次订单后可以对订单中所购买的商品进行评价。在浏览商品详情时，也可以查看到其他买家对这个商品的评价。

商品评论 ER 图（如图 3-6 所示）：

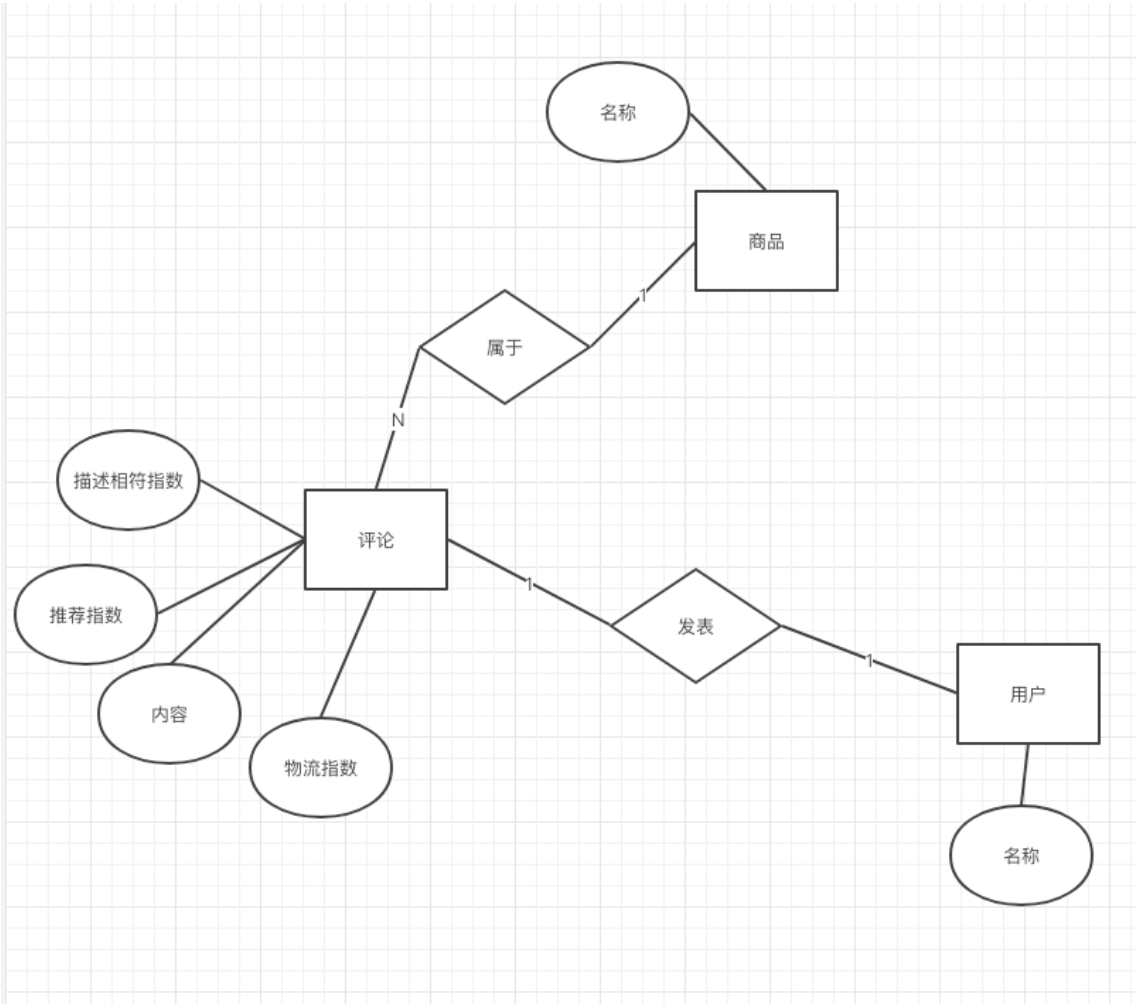

图 3-6 商品评论管理 ER 图

3.3.7 收货地址管理

主要是用户可以对收货地址进行地址的添加，编辑，删除

收货地址管理 ER 图（如何 3-7）：

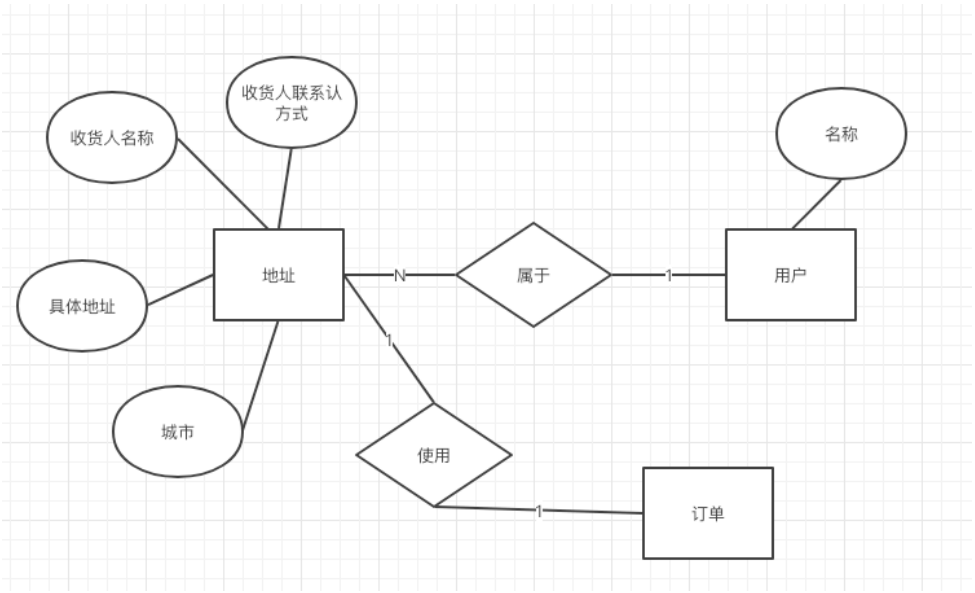

图 3-7 收货地址管理 ER 图

3.3.8 个人信息管理

对用户信息中的邮箱等附加信息进行编辑，或是修改密码等安全操作。个人信息 ER 图（如图 3-8 所示）：

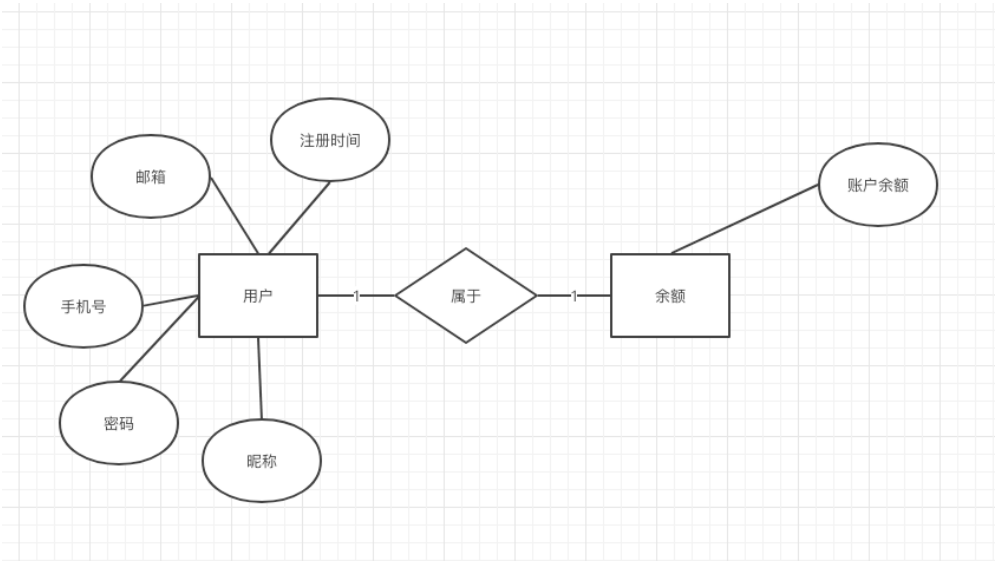

图 3-8 个人信息管理 ER 图

第 4 章 系统设计概述

4.1 系统体系架构

本系统采用 C/S 结构开发，即 Client/Server 架构，是客户端/服务器架构。这是大家熟知的且比较常用的软件系统体系结构，通过将各个不同的任务合理分配到 Client 端和 Server 端，降低了整个系统的通讯开销，因此需要安装客户端才可进行管理操作，用户也只能通过客户端进行操作。相对于三层体系结构（Client/Server 构架）是由逻辑上相互分离的表示层、业务层和数据层构成。表示层向客户提供数据，业务层实施业务和数据规则，数据层定义数据访问标准。三层体系结构中的核心是组件对象模型。它是软件系统体系结构，通过它可以充分利用两端硬件环境的优势，将任务合理分配到 Client 端和 Server 端来实现，降低了系统的通讯开销。大多数应用软件系统都是 Client/Server 形式的两层结构，由于软件应用系统正在向分布式的 Web 应用发展，Web 和 Client/Server 应用都可以进行同样的业务处理，应用不同的模块共享逻辑组件；因此，内部的和外部的用户都可以访问新的和现有的应用系统，通过现有应用系统中的逻辑可以扩展出新的应用系统。这也就是应用系统的发展方向。

C/S 结构具有以下优点：

（1） 开发比较容易，操作简便。

（2） 降低了系统的通讯开销。

（3） 用户体验更为一致。

4.2 系统模块设计

本 App 分为以下八个模块：登录注册模块，商品展示模块，商品收藏模块，购物车模块，订单管理模块，商品评论模块，收货地址管理模块， 个人信息管理模块。

4.2.1 登录注册模块

本应用的用户注册时，需要输入手机号，并收到验证码，在输入密码，验证通过，即为注册成功。可以使用手机号作为账号与设计的密码进行登录。

本应用的用户登录时，输入账号，即手机号，以及密码，验证通过后则完成登录。

用户在本应用的登录注册模块的流程图（如图 4-1 所示）：

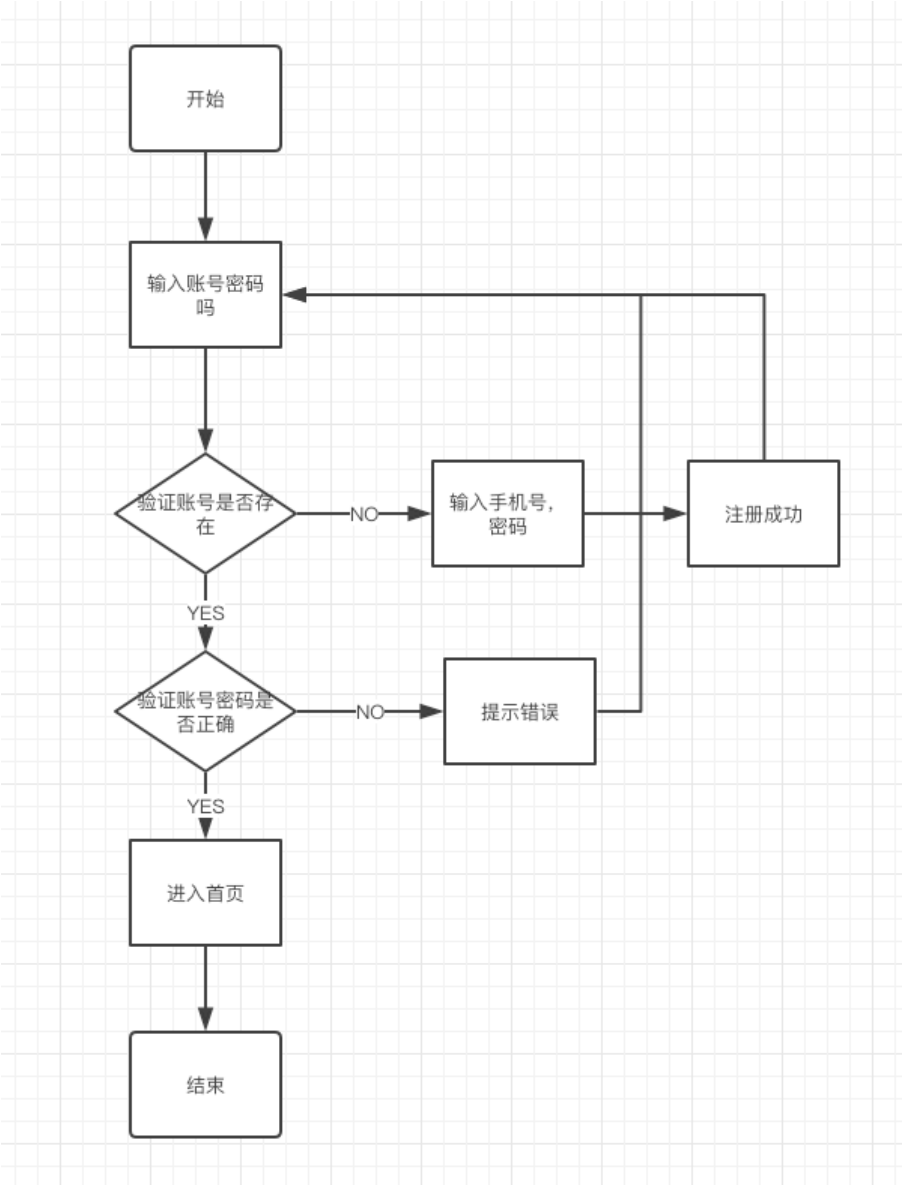

图 4-1 登录注册模块流程图

4.2.2 商品展示模块

本应用的用户进入精选页面后，可以浏览当前正在热卖的商品，以及在搞活动的商品。

本应用的用户进入分类页面后，可以通过滑动左侧商品类型进行选择类型，在右边可以选择具体到某一小类的商品，点击进入后可以看到在此小类下的所有商品，可以根据销量，价格等条件进行排序。

本应用的用户可以点击搜索按钮进入商品搜索页面，输入关键字后显示具体的商品。

当用户进入到商品的详情页后，可以查看商品的一些特色，同时可以选择查看关于商品的一些参数，以及其他用户对于该商品的评价。

用户在本应用中商品展示模块的流程图（如图 4-2 所示）：

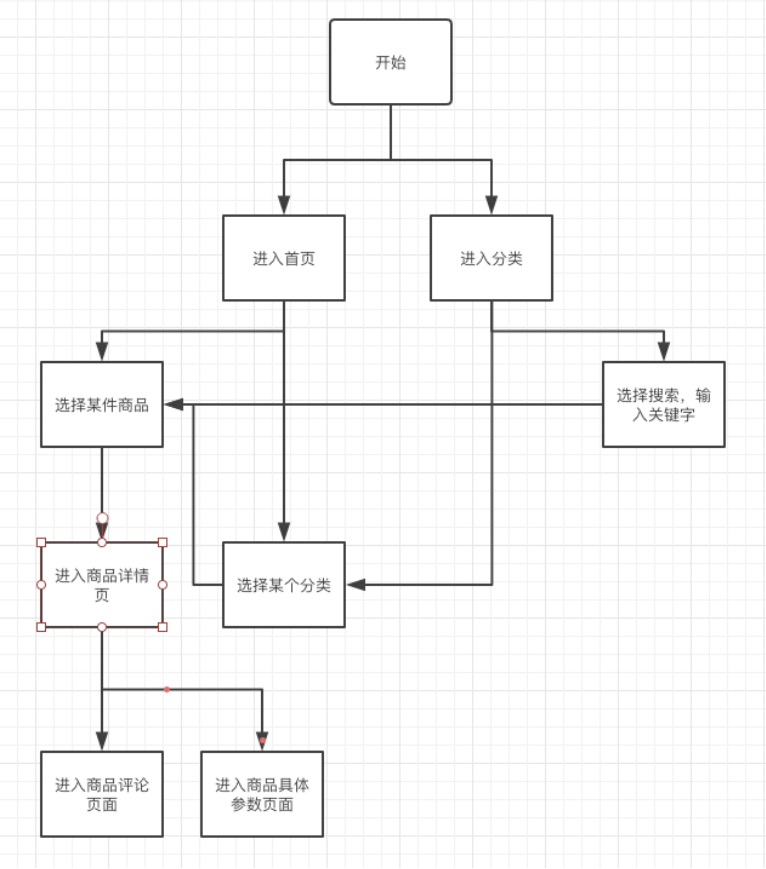

图 4-2 商品展示模块流程图

4.2.3 商品收藏模块

本应用的用户进入商品详情页，可以对该商品进行收藏或是取消收藏的操作。同时在个人中心中可以查看到本用户收藏的所有商品。并可以进入收藏商品的详情，对商品进行取消收藏操作。

用户在本应用中的商品收藏模块的流程图（如图 4-3 所示）：

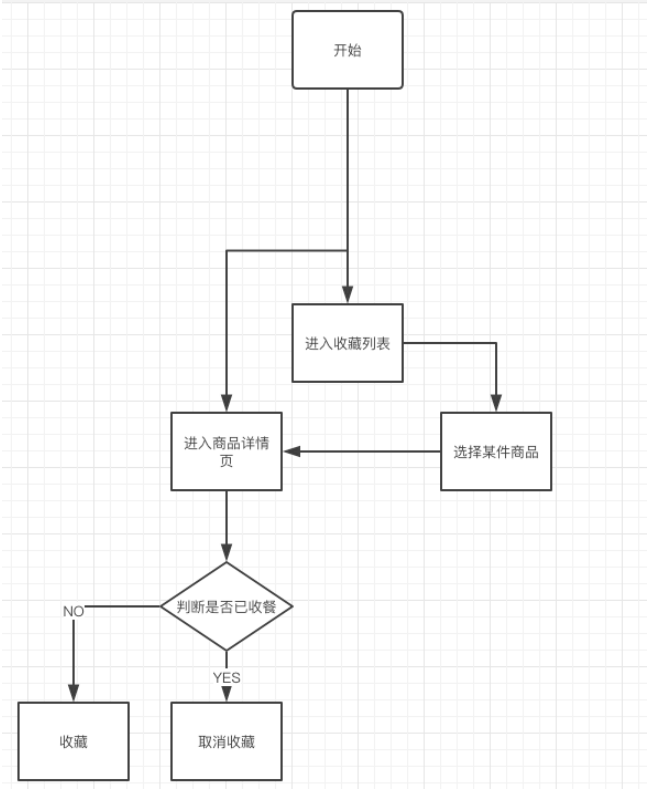

图 4-3 商品收藏模块流程图

4.2.4 购物车模块

本平台用户进入购物车页面可以查看当前所有已经加入购物车的商品，并可以修改各个商品的个数，可以触发结算操作。

本平台用户进入具体某一样商品的详情时，可以将这件商品加入到购物车，以便多件商品一起进行结算。

用户在本应用中的购物车模块的流程图（如图 4-4 所示）：

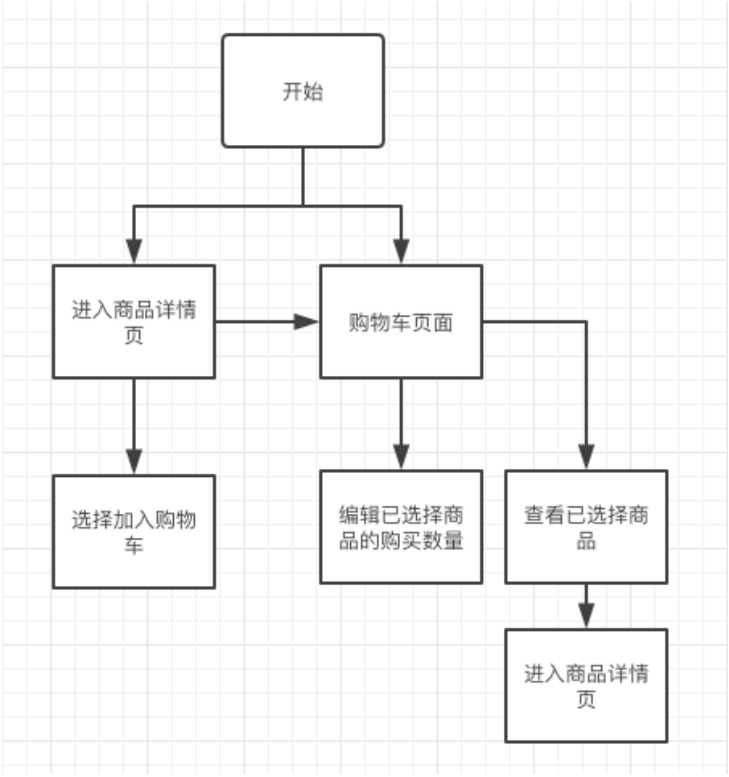

图 4-4 购物车模块流程图

4.2.5 订单管理模块

本应用的用户可以在商品详情页或是购物车页中选择好商品后发起新建订单。同时在个人中心的我的订单中可以查看所有类型的订单。对于未支付的订单可以进行支付，对于待收货的订单可以进行确认订单操作，对于待反馈的订单可以选择反馈，对于已经完成的订单可以选择退款。

用户在本应用中的订单管理模块的流程图（如图 4-5 所示）：

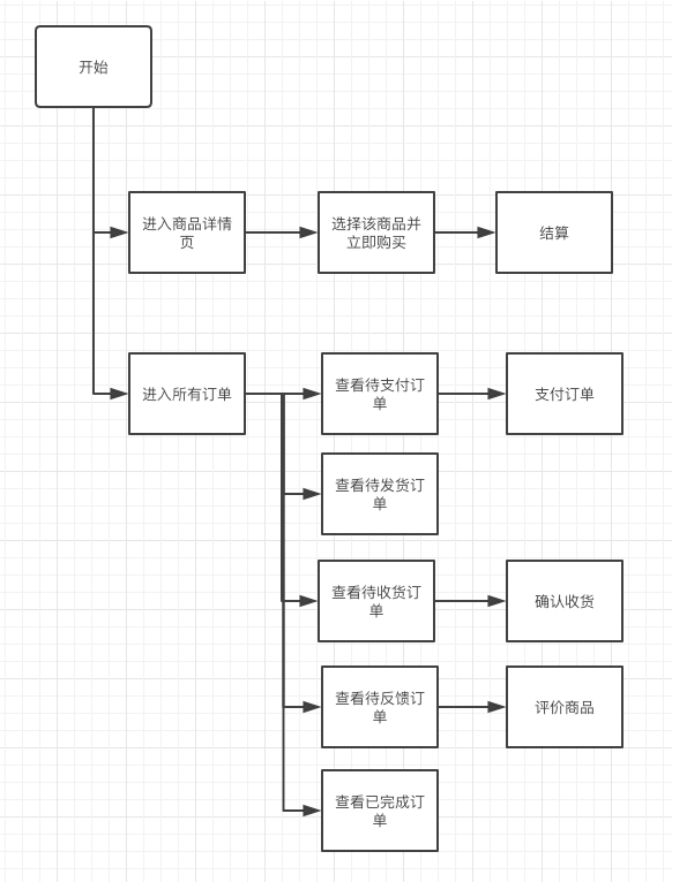

图 4-5 订单管理模块流程图

4.2.6 商品评论模块

本应用的用户在完成了一笔订单后，可以在订单列表要选择这笔订单，并对这笔订单中所涉及的商品，进行评论，该评论将会在该商品的详情页中展示出来。

用户在本应用中的商品评论模块的流程图（如图 4-6 所示）：

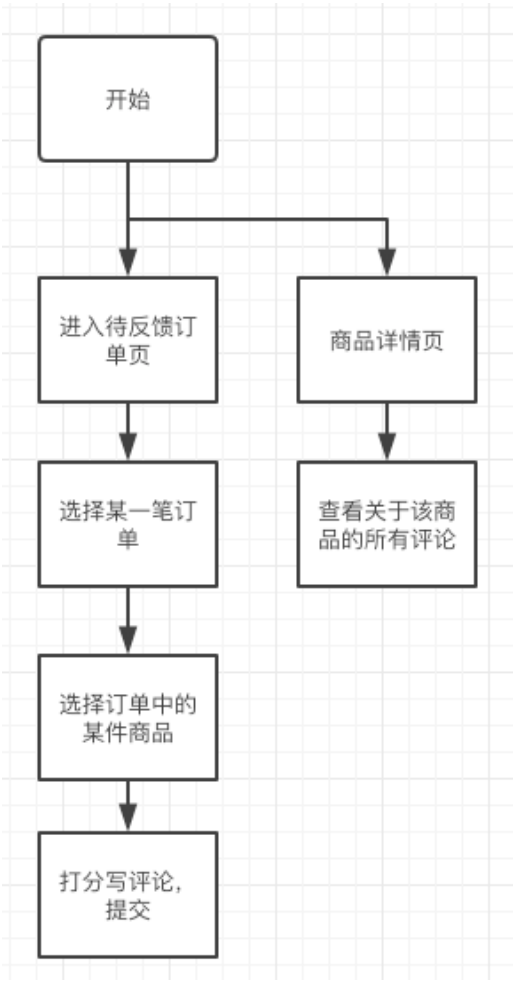

图 4-6 商品评论模块流程图

4.2.7 收货地址管理模块

本应用的用户在我的页面中可以选择我的地址，进入我的地址的列表页，在这里可以选择添加地址，或对已有地址进行编辑。同时可以把地址设置为默认地址，默认地址只允许有一个。

用户在本应用中的收货地址管理模块的流程图（如图 4-7 所示）：

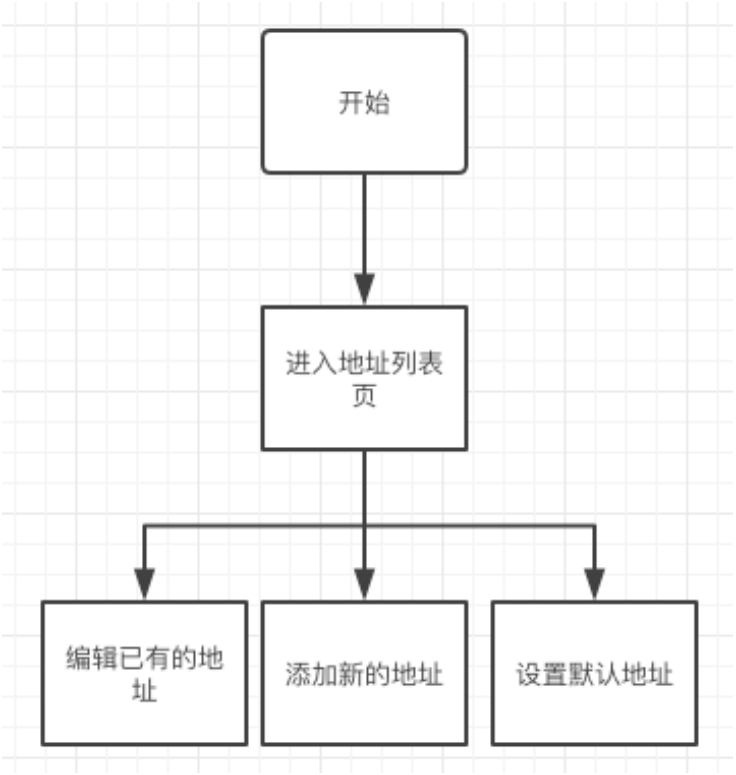

图 4-7 收货地址管理模块流程图

4.2.8 个人信息管理模块

本应用的用户在我的页面中可以选择个人信息，并对用户邮箱等信息进行编辑修改。也可以完成修改密码等安全操作。

同时在个人页面中可以对账户的余额中进行充值。

用户在本应用中的个人信息管理模块的流程图（如图 4-8 所示）：

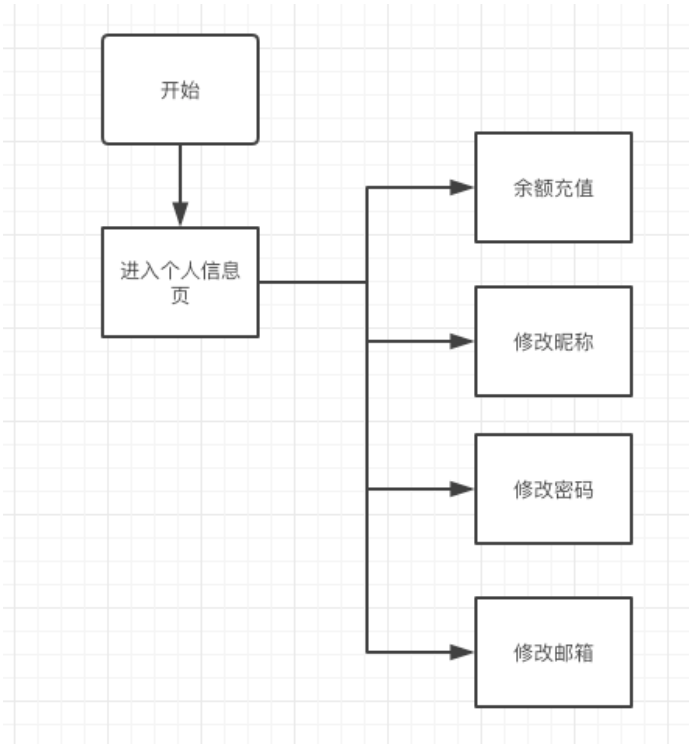

图 4-8 个人信息管理模块流程图

第 5 章 数据库设计与系统实现

5.1 数据库设计

本应用共有 10 张表，如下表所示：

表格 5-1 用户表

Field	Type	Comment
ID	Int(11)	主键
NAME	Vachar(36)	昵称
PASSWORD	Vachar(36)	密码
MOBILE	Vachar(11)	手机号
PORTRAIT	Vachar(200)	头像
EMAIL	Vachar(200)	电子邮箱
BLANCE	Double(10)	账户余额
TOKEN	Vachar(200)	用户标示
CREATED_AT	Datetime	创建时间
UPDATE_AT	Datetiem	修改时间

表格 5-2 用户地址表

Field	Type	Comment
ID	Int(11)	主键
DETAIL	Vachar(36)	详细地址
USERID	Int（11）	用户 ID
phoneNum	Vachar(11)	收货人手机号

ISDEFAULT	Int(11)	标记是否为默认地址
City	Vachar(200)	地址的省市区
CONTRACT	Vachar(200)	收货人姓名

表格 5-3 购物车表

Field	Type	Comment
ID	Int(11)	主键
DETAIL	Vachar(36)	详细地址
USERID	Int（11）	用户 ID
PHONENUM	Vachar(11)	收货人手机号
ISDEFAULT	Int(11)	标记是否为默认地址
CITY	Vachar(200)	地址的省市区
CONTRACT	Vachar(200)	收货人姓名

表格 5-4 类别表

Field	Type	Comment
ID	Int(11)	主键
NAME	Vachar(36)	类别名称
SUPERCATEGORY	Int（11）	父极类别
COVER	Vachar(11)	类别封面

表格 5-5 类别-商品关联表

Field	Type	Comment
ID	Int(11)	主键
CATEGORYID	Int（11）	类别 ID

GOODSID	Int（11）	商品 ID

表格 5-6 收藏表

Field	Type	Comment
ID	Int(11)	主键
USERID	Int（11）	用户 ID
GOODSID	Int（11）	商品 ID

表格 5-7 购物车表

Field	Type	Comment
ID	Int(11)	主键
CONTENT	Vachar(36)	评论内容
USERID	Int（11）	用户 ID
GOODSID	Int（11）	用户 ID
DESCRIPTIONMARK	Int(11)	描述相符程度
LOGISTICSMARK	Int（11）	物流满意程度
RECOMMENDMARK	Int（11）	推荐指数
USERNAME	Vachar(36)	用户昵称

表格 5-8 商品表

Field	Type	Comment
ID	Int(11)	主键
NAME	Vachar(36)	商品名称
PRICE	Double	商品价格
DESCRIPTION	Vachar(200)	商品描述

UNIT	Vachar（22）	商品单位
STOCK	Int（11）	商品库存
ORIGINAL_PRICE	Double	商品原价
STORAGE	Vachar(36)	商品储藏条件
BRAND	Vachar(22)	商品品牌
SHELFLIFE	Vachar(22)	商品保质期
POSTTAGE	Double	商品邮费
PICTURE	Vachar（22）	商品封面图
VOLUME	Int（11）	商品销量
MANUFACTURE	Vachar（22）	商品生产商
PACKING	Vachar（22）	商品包装方式
TYPE	Vacharr（22）	商品类型
FEATURE	Vachar（220）	商品特色描述
FEATUREPIC	Vachar（22）	商品特色图

表格 5-9 订单表

Field	Type	Comment
ID	Int(11)	主键
STATUE	Vachar(36)	订单状态
AMOUNT	Double	订单中商品总价
FARE	Double	订单运费
PAYMENT	Double	订单总价
CREATDATE	Vachar（22）	订单创建时间
PAYDATE	Vachar（22）	订单支付时间

SHIPDATE	Vachar(36)	订单发货时间
DEALDATE	Vachar(22)	订单完成时间
USERID	Int(22)	用户 ID

表格 5-9 订单-商品关联表

Field	Type	Comment
ID	Int(11)	主键
ORDERID	Int（11）	订单 ID
GOODSID	Int（11）	商品 ID
QUANTITY	Int（11）	数量

5.2 系统主要功能实现

5.2.1 登录注册模块实现

此模块（如图 5-1，图 5-2）中实现的主要功能为对用户账号的登陆与注册。用户可以使用手机号，与设置的密码进入本平台。同时新用户可以使用手机号，接受验证码，与设置的密码来进行注册。

关键技术 Storyboard

利用 Xcode 中内置的 Storyboard 进行界面的绘制，以及控件样式的设置。同时辅以少量代码实现业务逻辑。

关键代码：

```
//登陆按钮流程
@IBAction func confirmButtonClick(_ sender: Any) {
    //验证输入完整性
    guard checkStringAvailable(accoutTextField.text), checkStringAvailable(passwordTextField.text) else {
        HUD.showInfo("请输入完整信息")
        return
```

```
            }
            //发送网络请求
            HUD.wait()
            AuthertionAPI
                .userLogin(mobile: accoutTextField.text ?? "", password:
passwordTextField.text ?? "")
                .always {
                    HUD.clear()
                }
                .then { [weak self] user -> Void in
                    Default.Account.set(user.id ?? 0, forKey: .userId)
                    let animation = {
                        let oldState = UIView.areAnimationsEnabled
                        UIView.setAnimationsEnabled(false)
                        UIApplication.shared.keyWindow?.rootViewController
= MainTabBarController()
                        UIView.setAnimationsEnabled(oldState)
                    }
                    UIView.transition(with: self!.view,
                                      duration: 0.5,
                                      options: .transitionCrossDissolve,
                                      animations: animation,
                                      completion: nil)
                }
                .catch { (error) in
                    HUD.showError("登陆失败")
                }
        }
```

图 5-1 登录界面

图 5-1 注册界面

5.2.2 商品展示模块实现

此模块（如图 5-3， 图 5-4， 图 5-5，图 5-6， 图 5-7）中主要实现的对商品的不同维度的展示，通过添加多个入口，用户可以很轻易的找到一件商品。

关键技术：Instagram 的 IGListKit 框架，公司自制的 FormKit 框架

利用 IGListKit 来实现商品展示中的列表，通过定义不同的 item 来实现不同样式的 cell。

利用 FormKit 来实现简单样式的 Cell，利用不同种类的 Element 来组装成完整的 Cell。

关键代码：

```
/组装成 GoodsSectionItem
    public static func prepareGoodsItem(_ requireds: [GoodsCellRequired]) -> [GoodsSectionItem] {
        return requireds.map { GoodsSectionItem(data: $0) }
    }

//利用 FormItem 来组装 Cell
        let cartItems = cartMsg.flatMap {
            [
                FormItem(elements: [TextElement(text: NSAttributedString.attribute("包裹", .black, fontSize: 15))],
                         arrange: OneArrange(ArrangeCommon(ArrangeCommon.Direction.horizontal, ArrangeCommon.Alignment.center),
                                             location: OneArrange.AxisLocation.leading(20)),
                         height: 50,
                         separator: SeparatorLine(position: .bottom, margin: (20, 0), height: 0.5)),

                FormItem(elements: [TextElement(text: NSAttributedString.attribute($0.goods.name ?? "", .black, fontSize: 15)),
                                    TextElement(text: NSAttributedString.attribute("\($0.goods.price ?? 0.0) 元 * \($0.cart.quantity ?? 1)", .black, fontSize: 15))],
```

```
                arrange:
TwoArrange(ArrangeCommon(ArrangeCommon.Direction.horizontal,
ArrangeCommon.Alignment.center),
                                location:
TwoArrange.Location.symmetric(style: TwoArrange.Location.SymmetricStyle.siding,

spaces: (first: 20, last: 20))),
                height: 80,
                separator: SeparatorLine(position: .bottom, margin: (20,
0), height: 0.5)),
            FormItem(elements:          [TextElement(text:
NSAttributedString.attribute("邮费", .black, fontSize: 15)),
                                TextElement(text:
NSAttributedString.attribute("\($0.goods.postage ?? 0.0)元", .black, fontSize: 15))],
                arrange:
TwoArrange(ArrangeCommon(ArrangeCommon.Direction.horizontal,
ArrangeCommon.Alignment.center),
                                location:
TwoArrange.Location.symmetric(style: TwoArrange.Location.SymmetricStyle.siding,

spaces: (first: 20, last: 20))),
                height: 50,
                separator: SeparatorLine(position: .bottom, margin:
(20, 0), height: 0.5))
            ]
        }
```

图 5-3 应用首页界面

图 5-4 应用分类界面

图 5-5 应用商品列表界面

图 5-6 应用商品参数界面

图 5-7 应用商品详情界面

5.2.3 商品收藏模块实现

此模块（如图 5-8， 图 5-9）主要是给予用户一个对于某几件商品的收藏功能。用户可以在进入商品详情后选择收藏该商品。也可以在收藏商品列表中查看。

关键技术：RxSwift 框架。

利用函数式编程思想来实现这个功能，利用 RxSwift 将按钮的点击事件与发送是否收藏请求，进行绑定，同时来改变按钮的样式。

关键代码：

```
func objects(for listAdapter: ListAdapter) -> [ListDiffable] {
    return source
}
func listAdapter(_ listAdapter: ListAdapter, sectionControllerFor object: Any)
-> ListSectionController {
    return GoodsSectionController(delegate: self)
```

```
}
func emptyView(for listAdapter: ListAdapter) -> UIView? {
    return emptyLabel(message: "暂无数据")
}
```

图 5-8 应用商品收藏界面

图 5-9 应用商品收藏列表界面

5.2.4 购物车模块实现

此模块（如图 5-10）主要实现的是模拟现实生活中商品的一个购物车的功能，用户可以在商品的详情页面里，将该商品加入到购物车中。最后在购物车页面中实现商品个数的修改，查看，结算等功能。

关键技术：BEMCheckBox 确认操作框架。

利用 BEMCheckBox 来实现购物车中选择商品的确认。

关键代码：

```
static func prepareCommoditySectionItem(_ requireds: [Cart]) -> [CommoditySectionItem] {
    return requireds.map { CommoditySectionItem.init(data: $0) }
}
```

图 5-10 应用购物车界面

5.2.5 订单管理模块

此模块（如图 5-11， 图 5-12）中主要是实现的一个功能为用户在选择所需要购买的商品后，进行下单，以及下单后对订单的确认收货，反馈等操作。

关键技术：WMPageController 框架，FormKit 框架。

利用 WMPageController 来实现多个 tab 之间切换的页面。利用 FormKit 来实现不同类型中展示相似但又有少许不同的 Cell。

关键代码：

```
override func numbersOfChildControllers(in pageController: WMPageController) -> Int {
    return titleSource.count
}
override func pageController(_ pageController: WMPageController, titleAt index: Int) -> String {
```

```
        return titleSource[index]
    }
    override func pageController(_ pageController: WMPageController, viewControllerAt index: Int) -> UIViewController {
        return OrderListController(status: config(index))
    }
```

图 5-11 应用确认订单界面

图 5-12 应用订单列表界面

5.2.6 商品评论模块实现

此模块（如图 5-13）主要实现的是在商品详情页中的增加用户可以查看关于该商品的评论。同时在订单页面中确认收货后可以对该商品进行评论。

关键技术：iOS 原生计算高度功能，IGListKit。

利用 IGListKit 来实现列表的展示，同时通过 IiOS 中原生计算高度来计算 cell 的高度，以此来匹配评论内容不同而造成的不同高度。

关键代码：

```
//实现列表 item
    private func config() {
        source = comments.map { DataFactory.sectionItem.prepareGoodsCommentItem($0) }
        adapter.reloadData(completion: nil)
    }
```

图 5-13 应用订单列表界面

5.2.7 收获地址管理模块实现

此模块（如图 5-14， 图 5-15）主要实现的功能为用户可以管理自己的收获地址。其中包括对已有地址的编辑，添加新地址，删除地址。

关键技术：iOS 中的原生 tableView 控件。

用 tableView 来实现列表，通过实现原生左滑删除。

关键代码：

```
//实现删除
    func tableView(_ tableView: UITableView, titleForDeleteConfirmationButtonForRowAt indexPath: IndexPath) -> String? {
        return "删除"
    }
//实现列表
    func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell {
```

```
        let cell: AddressCell = tableView.dequeueReusableCell(withIdentifier:
AddressCell.reuseIdentifier) as! AddressCell
        cell.model = source[indexPath.row]
        return cell
    }
```

图 5-14 应用收获地址列表界面

图 5-15 应用编辑或添加地址界面

5.2.8 个人信息管理模块实现

此模块主要实现的功能为用户可以对自己信息的操作。包括对各个信息的编辑，以及对余额对充值。

关键技术：Storyboard，iOS 原生控件 tableview。

利用 Storyboard 进行界面的绘制，利用 tableview 的代理 delegate 进行业务逻辑的实现。

关键代码：

```
//进入修改界面逻辑
    override func tableView(_ tableView: UITableView, didSelectRowAt indexPath: IndexPath) {
        tableView.deselectRow(at: indexPath, animated: true)
        if indexPath.section == 0 {
            let controlelr = ModifyUserInfoController.initFromStoryboard(name: .mine)
            controlelr.userInfo = user
```

```
                if indexPath.row == 0 {
                    present(MainNavigationController(rootViewController:
controlelr), animated: true, completion: { controlelr.type =
ModifyUserInfoController.UserInfoType.name })
                } else if indexPath.row == 1 {
                    present(MainNavigationController(rootViewController:
controlelr), animated: true, completion: { controlelr.type =
ModifyUserInfoController.UserInfoType.email })
                }
            }
        }
```

图 5-16 应用个人信息界面

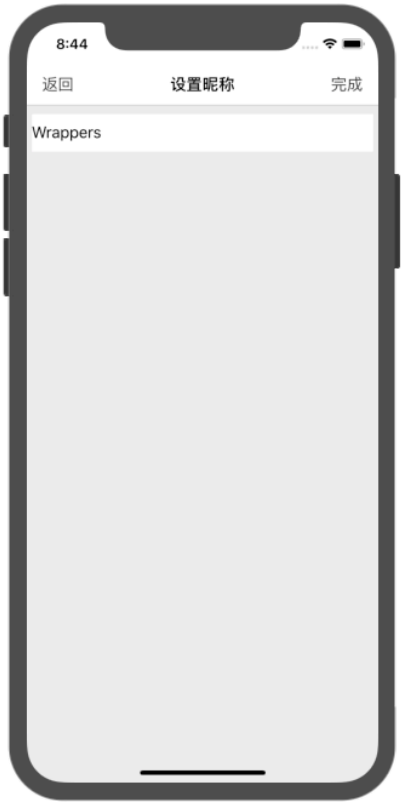

图 5-17 应用编辑信息界面

图 5-18 应用余额编辑界面

第 6 章 系统测试与维护

6.1 系统测试

本系统测试是采取先测试代码中的单元测试。再对单个模块的功能，合格后在进行下一模块的测试，逐一完成模块粒度的单元测试，然后对整个 App 所有流程进行进行系统测试，在测试过程中主要采用以下手段进行测试：

（1）进行功能性测试，进行各种不同的操作检查系统反馈情况；

（2）采用 debug 功能调试代码，或利用 Reveal 进行界面元素的审查，以此来排查流程中蕴含的错误信息。

在对本 App 进行测试时，同时关注了才本 App 运行时的心流量、电量、CPU、GPU、Memory。本 App 在测试过程中均为出现耗费大量系统资源的情况。

同时，针对 iOS 设备不同屏幕之间适配的问题，我将 App 运行到各个不同屏幕尺寸的设备上，进行测试。尤其关注了 iPhone5s 这类小屏幕设备到兼容，根据测试内容，进一步对 App 在不同设备间的适配进行优化。

针对 App 场景下，一些异常场景的考虑以及弱网络测试。这里的异常场景就是中断，来电，短信，关机，重启等。而弱网测试是 App 测试中必须执行的一项测试。包含弱网和网络切换测试。需要测试弱网所造成的用户体验，重点要考虑回退和刷新是否会造成二次提交。需要测试丢包，延时的处理机制。避免用户的流失。

针对 App 界面操作层面的测试：现在 app 产品的用户都是使用的触摸屏手机，所以测试的时候还要注意手势，横竖屏切换，多点触控，事件触发区域等测试。

以及 App 独有的特性：因为 App 是客户端，则必须测试安装、更新、卸载。除了常规的安装、更新、卸载还要考虑到异常场景。包括安装时的中断、弱网、安装后删除安装文件，更新的强制更新与非强制更新、增量包更新、断点续传、弱网，卸载后删除 app 相关的文件等等。

针对系统后台接口的测试，主要使用的 postman 工具，来模拟请求的返送，判断返回的结果是否为期望值。同时利用 Chrome 的开发工具来查看各个接口返回的速度。

6.2 系统维护

本 App 当下用户相对较少，操作逻辑也较为简单。往后 Swift 语言的发展可能会对系统的维护造成一些麻烦，但现在迁移工具比较，这将不会消耗很多时间。同时系统后台对于用户量大，高并发大操作实现的并不好，日后可能还需要这个方面进行不断优化，以及后台操作的 SQL 优化等方面进行适当优化。

开发容易维护难。对于 App 来说，只有不断地更新内容，才能保证 App 的生命力，否则 App 不仅不能起到应有的作用，反而会对自身形象造成不良影响。如何快捷方便地更新 App 内容，提高更新效率，是很多 App 面临的难题。现在网页制作工具不少，但为了更新信息而日复一日地编辑网页，对信息维护人员来说，疲于应付是普遍存在的问题。 内容更新是网站维护过程中的一个瓶颈。在未来的发展中，需要有计划的去完成一个对于 App 内容管理的网站，有助于对 App 内容的及时更新。

结束语

经过一个多月的坚持，终于完成了基于 iOS 平台的生鲜购物 App 的设计开发以及论文的撰写等工作。在这一个多月里，无论是开发上还是界面设计上，我都遇到了一些之前没有尝试过的挑战。通过不懈查询资料，以及不断的尝试，克服了这些大大小小的挑战。

在这一次完成毕业设计的过程中，我觉得我使自己了解到了很多以前未曾关注的知识面。从应用设计的初期时的界面设计，我通过不断的查询资料，学习了界面设计工具 Sketch 的基本使用，以及利用 Sketch 进行的原型的设计，与具体界面的交互逻辑。这是算是在这次毕业设计过程中的一次意外的收获。另一方面对于我的挑战在于后台的编写。我本身是在公司担任 iOS 开发，后台包括 Sql 语句并不是很熟悉。这次在后台的技术之所以选择了 Egg 作为后台框架而不选择更为保险的 Java 后台的方案的原因主要是本人希望通过这次毕业设计多学习一些 Js 的知识。同时这个决定，在我实际开发中也为了造成很不小的困扰，尽管 Node.Js 搭建后台非常方便，且拥有很多实用的脚手架工具，对语言的不熟悉，以及对框架结构的不了解，让我的后台编写进度比较缓慢。所幸勤能补拙，在经过不断碰壁，后半部的后台的开发时已经较为流畅了，对 Js 这一门语言也有了自己新的认识。当然过程中还有其他的一些小插曲，例如 Swift 的版本问题等，所幸都在老师和同学的帮助下，完整的解决了。

在这一次前后台独立开发的过程中，我算是完完整整的体验了一些软件开发所有的流程。每一份文档，每一行代码都是通过双手敲出来的，这次体验对于我日后再公司中的工作会产生巨大的帮助，我对项目开发会有一个整体完整的的大局观，而不再是之前狭隘的只关注自己所负责的 iOS 端的功能。同时在技术方面通过这次使用 Js 也为我之后朝大前端方向学习增加了一些信心。

最后，在这次毕业设计中，我的到的是一个完整的软件开发流程，以及让自己勇于接触未知领域的勇气。

致谢

人生就是一个关于离别的漫长故事．纵然你此身犹在，却已于某些地方、某些人处,和你的往昔时光永诀了.大学作为人生的一部分,亦是这样的一段故事.大学四年，俯仰之间．“问道”、“学术”于此，我不得不感叹“今我睹子之难穷也，吾非至于子之门则殆矣”。

这一次毕业设计的开发与撰写文档，是对于我这个软件工程学生大学四年学习的知识的一次集中的体现。不但体现了我大学四年的学习成果，更让我发现和明白了自己的不足与在知识面上的狭隘。本 App 的开发是我对于这段时间内，努力付出的成果。在这一整个过程中，尽管大多数时候都比较顺利，但有时也会遇到一些重难点的问题，顿时间内无法克服，所幸我的指导老师孙奕鸣、周围同学的帮助下，这些重难点的问题被我逐一攻破了，最终本次毕业设计落下了完美的帷幕。

最后感谢父母多年来在各种方面给予我的物质或是精神上的支撑，感谢在我遇到困难给予我帮助，给予我信心的人，是你们的付出才能成就今天的我。同时我要向我的指导老师孙弈鸣以及帮助过我的同学们表示诚挚的谢意，同时也要在此向整个浙江科技学院软件工程的所有授课老师表示谢意。感谢所有任课老师们在四年里对我的殷切教导与鼓舞，也感谢所有一起走过四年的同学们，感谢我在浙江科技学院认识的所有朋友，与你们在一起的这四年，是我人生中度过最难忘的四年。能与你们在浙江科技学院相遇是我人生之一大幸事。相聚总有分离时，愿我们软件工程 18 届的同学在未来的人生路上再能重逢,志在星辰大海的有梦青年。

参考文献

[1] 田妞。 WWDC 2013 亮点汇[J]. 中国质量万里行。 2013(07)

[2] 黄佳星，王晶，沈奇威。 基于 Android 的移动互联网应用框架方案[J]. 电信工程技术与标准化。 2012(08)

[3] 曹森，苏贵斌。 软件开发中的设计原则[J]. 软件导刊。 2012(01)

[4] 罗军舟，吴文甲，杨明。 移动互联网：终端、网络与服务[J]. 计算机学报。 2011(11)

[5] 陈子涵，吴明晖，应晶。 基于 MDA 的移动应用开发框架[J]. 计算机工程。 2011(18)

[6] 夏浩波。 单例模式的设计与应用[J]. 电脑开发与应用。 2011(01)

[7] 黄金国，罗震。 手机应用程序开发架构的研究[J]. 计算机工程与科学。 2010(11)

[8] 张琳琳，应时，倪友聪，赵楷，文静。 一种软件体系结构关注点分析方法[J]. 计算机学报。 2009(09)

[9] 张伟，梅宏。 一种面向特征的领域模型及其建模过程[J]. 软件学报。 2003(08)

[10] Matthew J. Evans,Graeme Clemens,Christopher Casey,Matthew J. Baker. Developing a mobile app for remote access to and data analysis of spectra[J]. Vibrational Spectroscopy . 2014

[11] Pocatilu,Paul. Developing an M-Learning Application for iOS[J]. Informatica Economica . 2013 (4)

[12] T.R. Gopalakrishnan Nair,R. Selvarani. Defect proneness estimation and feedback approach for software design quality improvement[J]. Information and Software Technology . 2011 (3)