论文原创性声明

本人郑重声明：所呈交的毕业论文，是在导师的指导下，独立进行研究所取得的成果，所有数据、图片资料真实可靠。除文中已经注明引用的内容外，本论文的研究成果不包含他人享有著作权的内容。对本论文所涉及的研究工作做出贡献的个人和集体，均已在论文中以明确的方式标明。本论文的知识产权归属培养单位。

本人签名：　　2023年 4 月 2 日

论文版权使用授权书

本论文“基于 Spring+Vue 的吾爱理财平台的设计与实现”是本人在校期间所完成学业的组成部分，是在江苏师范大学科文学院教师的指导下完成的，因此，本人特授权江苏师范大学科文学院可将本毕业论文的全部或部分内容编入有关书籍、数据库保存，可采用复制、印刷、网页制作等方式将论文文本和经过编辑、批注等处理的论文文本提供给读者查阅、参考，可向有关学术部门和国家有关部门或机构呈送复印件和电子文档。本毕业论文无论做何种处理，必须尊重本人的著作权，署明本人姓名。

作者签名：　　指导教师签名：

2023 年 4 月 6 日　　2023 年 4 月 6 日

基于 Spring+Vue 的吾爱理财平台的设计与实现

摘　要

截止至 2023 年 6 月底，我国投资者数量高达 9145.40 万，上半年累计为投资者创造收益 4172 亿元，个人投资者风格逐渐趋于保守，稳健性个人投资者占比 35.51%。由于新冠疫情的影响，投资市场长期处于低谷，短期回撤力度偏大，导致许多个人投资者在投资市场中找不到方向，甚至迷茫。如何选择投资产品，在什么时间段选择什么方向的投资策略成为了一个摆在个人投资者面前的严峻问题。

随着社会的发展，社会的方方面面都在利用信息化时代的优势。互联网理财是一种新事物，就让我们在应用中学习，逐步提高自己的理财能力，理财贯穿于我们生活中的各个细节中，随着网络与网民生活的日益融合，互联网理财应该会成为网民生活中的一个热点。

本吾爱理财平台借助互联网的优势开发一个关于市面上常见理财产品的交互平台，促使理财平台变的更加系统化、有序化。致力于为热爱投资，对投资感兴趣的新个体提供产品介绍，类型分析，能够帮助普通投资者更好的利用闲散资金，合理组合资产实现产品收益。

该论文有图 44 幅，表 25 个，参考文献 13 篇。

关键词：理财　投资　互联网　资产

Design and Implementation of My Love Financial Platform

Abstract

By the end of June 2023, the number of investors in China has reached 91.454 million. In the first half of the year, it has accumulated 417.2 billion yuan of income for investors. The style of individual investors has gradually become conservative, and the proportion of stable individual investors is 35.51%. Due to the impact of COVID-19, the investment market has been at a low point for a long time, and the short-term pullback is too strong, resulting in many individual investors being unable to find direction in the investment market, or even confused. How to choose investment products and what direction to choose at what time has become a serious problem for individual investors.

With the development of society, all aspects of society are taking advantage of the information age. Internet financial management is a new thing. Let's learn from the application and gradually improve our financial management ability. Financial management runs through every detail of our life. With the growing integration of the Internet and Internet users' lives, Internet financial management should become a hot spot in Internet users' lives.

This "My Love Financial Management" platform develops an interactive platform about common financial products on the market by virtue of the advantages of the Internet, which makes the financial management platform more systematic and orderly. It is committed to providing product introduction and type analysis for new individuals who love investment and are interested in investment, which can help ordinary investors make better use of idle funds and reasonably combine assets to realize product income.

This paper has 44 figures, 25 tables and 13 references.

Key words: Financial Investment Internet Assets

目　录

图清单

续图清单

表清单

1 绪论

1.1 理财平台的现状与发展

随着网络化的发展，私人投资理财工具逐步向网络发展。个人足不出户，即可通过手机、电脑等设备进行投资理财。在投资理财网络化方面，传统的股票、基金、保险服务的购买和交易都已经可以在网上进行，此外，新型的理财产品在网上也已经越来越多，人们可以通过网络进行虚拟贵金属买卖、期货买卖、网络借贷等。从互联网理财用户需求来看，互联网理财产品具有的低门槛、互联网理财高收益和高流动性特点，贴合大众理财需求。

未来，随着大数据、人工智能等技术不断成熟，互联网理财产品优势更加凸显。伴随着中国居民财富的持续积累，全市场对于规范、健康和全面丰富的财富管理的呼声不断增加，众多机构也将未来展业的方向专注在互联网财富管理的蓝海市场。

1.1.1 国外发展现状

欧美等西方国家的私人银行业务这个家族已有 100 多年的历史。最初，瑞士银行业转向了极其富有的客户（甚至数亿美元），提供私人客户专属的一对一服务，如匿名存款服务（目前是瑞士银行业最特殊的服务），提供财务咨询、投资咨询，甚至安排客户参观获得医疗、安排旅行和住宿、安排客户的孩子上贵族学校等。简而言之，私人人民银行为客户提供即时、一对一、全面、全面的服务客户的资产和个人数据应完全保密。

许多外资商业银行的个人金融业务既面向高端富人阶层，也面向公众富人阶层，分别提供私人银行服务和 VIP 金融服务。例如，美林致力于数百万美元百万富翁提供私人银行服务，如财务咨询和信托，而资产不足 100 万美元的百万富翁富有的客户接受以优先和折扣为特征的 VIP 融资业务。

1.1.2 国内发展现状

相较于国外来说，互联网理财已经成为我国 90 后理财首选，报告显示 88%的 95 后理过财，互联网理财[1]购买率超过 80 后!并且，有超过 16%的 95 后预期收益率在 20%以上，超过股神巴菲特的投资收益率。互联网理财拥有高于其他无固定期限理财产品的收益率，通常是 3-5 倍，跟银行存款相比更不必说了。

另外，互联网理财产品很多是没有出资门槛的。传统理财中，一般都在几万元以上，门槛上现在很多想理财的人都很害怕，网络理财几乎没有门槛，100 元即可出资，任何

人都可以理财，与传统理财相关，互联网理财倾向于大众和中小企业。当然，最重要的还是操作方便，只需要在手机上动动手指就可以完成操作。

当然，市场蓬勃发展的同时，危害也正在蔓延。目前我国网民群体中青年学生占比最多，占 21%，校园贷、虚假理财投资等网络金融诈骗类案件，暴露出消费者金融常识的缺乏。

1.2 吾爱理财平台的研究内容

本课题研究的主要内容是针对现有理财平台网站存在的功能简单、产品种类不多、用户使用感差等多方面的问题，整理归纳问题，寻找改进的方法然后逐个击破。首先是平台安全性的问题，该平台在用户未开通个人账户的前提下不允许购买理财产品只能够预览理财产品与了解产品类型详情。对于一个面向于个人的理财平台，拥有一个查看个人中心的区域是很重要的，投资者可以在此查看个人余额和订单详情，并且搭载了数据图表随着数据的动态显示。其次是后台管理权限问题，对于管理员，多数网站缺乏权限管理，只设置了管理员一种权限类型，本平台通过权限判断，赋予管理员不同权限的功能执行，比如低权限管理员只能去修改其下级权限的管理员信息。并且平台带有规范的后台管理系统是必要的，后台管理首页能够实时显示本地天气信息，更重要的是各类统计数据的查看，通过数据图表的显示让数据更加的可视化。

本课题就“吾爱理财”平台展开研究，网站提供给前端用户的主要功能有用户将感兴趣的理财产品加入预购、用户提交单个产品订单、用户从预购中心批量形成订单、用户个人余额的充值、用户查看理财产品详情、播放产品简介视频、用户浏览各类理财产品、用户提交订单支付等多种操作，网站也给用户提供了友好提示，当用户操作错误或超出网站操作范围时，会弹出提示框提醒用户。网站建有规范的后端管理系统，对于前端用户操作，后端系统会有信息实时的回显，方便管理员查看并管理管理员信息、用户信息、权限信息、理财产品信息、预购中心信息、订单信息等，管理员要及时查看并处理用户的订单方便给用户结算金额，后端加入的 ECharts 数据可视化，使繁杂的数据通过简洁的图表形式展示出来，方便管理员随时跟踪网站的运行情况。

1.3 吾爱理财平台的研究目的和意义

通过基于 Spring[14]+Vue 的吾爱理财平台的主要客户是理财产品消费者，相应的系统功能也分为用户功能和管理员功能，用户功能主要是对理财产品的浏览查询和购入，管理员功能主要是对理财产品的管理，用户的管理和订单的管理。

互联网理财是一种新事物，让我们在应用中学习，逐步提高自己的理财能力，理财贯穿于我们生活中的各个细节中，随着网络与网民生活的日益融合，互联网理财应该会成为网民生活中的一个热点。在理财产品的选择上一般是综合权衡各种资产存在方式的流动性、安全性、风险性，根据自己的需要和风险偏好进行选择。

因此本项目制定一个基于 Spring+Vue[3]的理财平台系统，并从用户模块、后台管理模块、预购模块、订单模块、支付模块、门户模块、角色权限模块及密码修改模块来完善理财平台管理系统。

1.4 本章小结

本章对本课题进行了较为详细的说明，首先是对国内外的理财平台的发展进行了简单对比，虽然国内理财平台的发展历史不如国外，但是我们的理财人数增长率远大于国外，且理财平台发展仍有巨大潜力，越来越多的人参与投资，学习如何理财，让理财概念融入生活。国内的理财平台也在日趋完善。本“吾爱理财”平台就是在腾讯理财通，支付宝理财的基础上，构建一个界面更加美观，操作更流畅，功能更完善，用户体验感更好的理财平台网站，旨在帮助普通投资者更好的了解理财产品，根据自身实际情况选择更加适合自己的投资方向，做到不盲目投资，逐渐去形成一种优秀的理财概念。

2 本吾爱理财平台的分析

2.1 可行性分析

本章节对吾爱理财平台的技术、经济、操作等方面都做了可行性分析。

2.1.1 技术可行性分析

本吾爱理财平台是以 Java 为开发技术，它具有简单方便、面向对象、稳定等特点；使用流行框架 Spring Boot 能够快速搭建项目，使项目能够独立运行，无需外部依赖 Servlet 容器，极大的提高了开发效率；前端页面使用 Element UI 框架搭配 Vue[9]框架，Vue 是一个渐进式开发框架，非常适合用于构建用户界面；Oracle[12]数据库的使用为开发者提供了基于角色分工的安全保密管理，且其提供了新的分布式数据库能力，可通过网络较方便的读写远端数据库的数据。通过这些技术构建的网站，可以更快更好地进行搭建并完善网站，对后期网站的维护也大有益处。因此，本网站具有技术可行性。

2.1.2 经济可行性

要开发一个项目那必然要考虑它的经济可行性问题，首先要预估项目的开发成本及效益。本吾爱理财平台是使用 Java 开发语言，网站的规模适中，开发周期短，因此，在项目的开发阶段不会投入太多资金。网站投入使用后需要不断的资金维护网站的稳定运营，但网站在使用期也会有资金收益回拢。因此，本“吾爱理财平台”具备经济可行性。

2.1.3 操作可行性

本平台采用菜单式，实现用户与数据库交互，界面简洁友好，方便操作，用户只需了解购买流程和业务调查就可以轻松上手，不需要掌握数据库等相关知识。因此，操作上完全可行。

2.2 需求分析

根据对国内外理财平台的发展现状的研究，分析了现如今庞大基数的投资人群的概念缺陷，并结合了用户的需求，以确定系统和受众人群为目标，总结得到本吾爱理财平台在功能设计和性能要求方面的需求[16]

2.2.1 功能需求

(1)前台界面模块

作为普通用户的购入前的门户页面，主要分为首页，股票，基金，债券，银行储蓄，保险，黄金，以及个人中心和登录导航的设计，其中每个理财产品模块包含了对每个理财产品的入行指南，产品介绍视频，基金经理等区域设计。其中在首页搭载产品动态数据图表，最新理财资讯。在理财产品模块可进行理财产品购买，添加到预购中心，若想查看理财产品详情可以点击进入查看；也可以直接对某支产品进行支付形成订单，并选择支付的方式和服务指数。

(2)用户中心模块

通过前台首页登陆按钮进入登录页面，唯有用户登陆后才可以进行理财产品的购买和个人信息的修改和查看，登录采用了输入用户名，密码，以及滑动验证的方式，提高了用户登陆的安全性。 登录成功后可以进入用户个人中心，大致分为两层，第一层左侧显示该用户的收入、支付、总资产、余额的饼状图表，右侧显示用户已完成的所有订单；第二层显示订单的详情，其中左侧显示用户在不同时间的支出和收入，右侧表格显示该用户购买的所有理财产品，并做到去除重复产品。

(3)后台管理模块

总体分为用户管理，理财管理，订单管理，公告管理，意见反馈和系统管理等小模块。其中用户管理中包括用户列表，角色管理，用户新增等子模块，能够实现对用户角色权限的管理，以及新用户的添加。在理财管理中可以通过数据库中对理财类型的字段遍历查询得出股票，基金，债券，银行储蓄等理财产品的分组管理订单管理中首先显示所有订单列表，其次可根据普通用户对该订单的支付情况分为已支付，已完成，已取消订单，并且管理员可以对所有订单的部分信息进行修改与删除在公告管理中可对前台门户页面首页的公告进行更新以及图表数据的管理。

(4)预购中心模块

类似于商城网站的购物车模块设计，可以将用户想要的某项理财产品添加到预购中，然后可以继续选购，继续向预购中添加理财产品，用户可以在预购中清楚的看到自己所选的理财产品和数量以及购买所需的金额并进行总价计算，并且可以批量删除预购中不想要的理财产品，当用户选购结束后，可提交订单进行结算，结算时采用了批量添加的功能，即在后台管理或者用户中心可以查询出用户购买的具体产品细节。

(5)订单模块

分为订单简介和订单详情子模块设计，可在首页点击购买理财产品按钮后进入订单简介，填写购入数量以及支付信息的确认。在订单模块中如果点击取消订单，该订单记录将会发往后台已取消订单中。当成功支付订单后，该订单则将发往后台已支付订单中。当后台管理员审核后该订单状态改为已完成订单。若对订单信息不够了解，或者根据产

品的图片进入相对应的详情页面确认。

(6)支付模块

根据订单模块传递来的订单信息或者在用户个人中心进行个人余额的充值，点击按钮跳转至支付页面根据订单总价以及支付的方式进行支付，支付成功后可在首页个人中心查看已完成订单，管理员可进入订单管理查看细节。

2.2.2 性能需求

本吾爱理财平台是面对所有人，网站的兼容性高，稳定性好，在各类电脑上都能够稳定运行，也适应多数主流浏览器，因此，本网站满足性能需求。

2.3 框架介绍

本章节是针对吾爱理财平台的设计与实现做的技术分析，并得到相应的结论。

2.3.1 Spring Boot

在 Spring Boot 是由 Pivotal 团队开发，它有明确的目的——简化应用 Spring 开发项目。最突出的特点是配置方式，简化了 Spring 应用各个方面的配置。另外 Spring Boot 能够集成大量的框架，解决了之前很重要的项目之间包的版本依赖和稳定性问题。

采用 Spring Boot[4]框架,系统整体性能满足实际要求,特别是应用了前后端分离、前后端负载均衡,极大提高了系统运行的稳定性和可靠性[6]。使用 Spring Boot 集成框架,围绕 Web 进行架构,框架采用了表现层、业务逻辑层、持久化层三层体系[6]。

Spring Boot 还为大中型项目提供了经常用到的非业务功能型的特点：健康状态检测，外部配置、指标、安全等；在项目中不再推荐使用 XML 的方式[7]。

Spring Boot 它旨在简化新 Spring[10]应用程序的初始构建和开发过程。该框架使用特定的方式进行配置，因此开发人员不再需要定义模板化配置。这样，Spring Boot 致力于成为迅速发展的应用程序开发领域的领导者

2.3.2 Swagger

对于后端来说，编写接口文档即费时费力，还会经常因为没有及时更新，导致前端对接时出现实际接口与文档不一致。而且手写接口文档还容易出错，而 Swagger 很好的解决了这个痛点。

Swagger 是一个规范和完整的框架，用于生成、描述、调用和可视化 RESTful 风格的 Web 服务。可用于：1.接口的文档在线自动生成 2.功能测试。

2.3.3 Element UI

Element UI 是一套为开发者、设计师和产品经理准备的基于 Vue 2.0 的桌面端组件库，包含了一套设计原则、组件和组件，还·提供如 Axure 组件、Sketch 模板的设计资源。

Element UI 由饿了么前端开源维护，当前版本是 2.15.10，是最流行的 Vue 框架之一。

2.3.4 Vue.js

Vue.js[2]是一个用于构建用户界面的渐进式 JavaScript 框架，使用 Axios 插件封装 Ajax 进行数据交互，实现数据持久化。

Vue.js 是一个轻巧、高性能、可组建化的 MVVM 库，同时拥有非常容易上手的 API；Vue.js 是一个构建数据库驱动的 Web 界面的库；Vue.js 是一套构建用户界面的渐进式框架，与其他重量级框架不同的是，Vue.js 采用自底向上增量开发的设计。Vue 的核心库只关注视图层，非常容易学习，易与其他库或已有项目整合。另一方面，Vue[6]完全有能力驱动采用单文件组件和 Vue[15]生态系统支持的库开发的复杂单页应用。简而言之，Vue.js 的目标是通过尽可能简单的 API 实现响应的数据绑定和组合的视、图软件。核心是一个响应的数据绑定系统。Vue.js[8]的特性有：轻量级的框架；双向数据绑定；指令；插件化。

2.3.5 ECharts

数据可视化是利用人眼感知能力和人脑智能，利用图形、计算机视觉等日益成熟和完善的手段，结合数据的特点，以易于理解的方式，对数据交互进行可视化表达，传输有用信息是国内外数据可视化领域研究人员的主要方向之一[13]。

ECharts[7]是一个纯 JavaScript 的开源的可视化图表库，涵盖了各个行业的图标，它提供了更直观、生动、可个性化定制的数据可视化图表，能够满足各种需求。ECharts 兼容了当前绝大部分的浏览器及多种设备，可以满足本网站的性能需求。

2.4 本章小结

这一章节从三个方面对本吾爱理财平台做了分析。先是从网站可行性方面对本吾爱理财平台做了经济可行性、技术可行性和操作可行性这三个方面的分析，分析结果显示本网站可行。而后从网站的功能需求和性能需求两方面是对本网站做了需求分析，较详细的说明了本网站的功能需求，简要概括了性能需求。最后，对网站所使用的技术框架做了简单的概念讲解。

3 本吾爱理财平台概要设计

3.1 网站功能和数据设计

在对本“吾爱理财平台”进行编码实现之前，必须要对网站进行整体功能分析。需要对其进行整体分析。这里，将本网站划分为两个部分，即前台和后端管理系统，这两个部分每一个里面都包含了好几个模块，需要对这些模块之间的关联关系进行整理。通过对网站功能进行整体功能分析，并设计出功能结构图，可以使本网站前后端功能明确，架构层次清晰明了；在之后编码实现时，也能够分模块进行，快速上手。后期对网站的维护工作也会根据模块划分来进行，提高工作效率。本“吾爱理财”平台的整体功能结构见图 3-1。

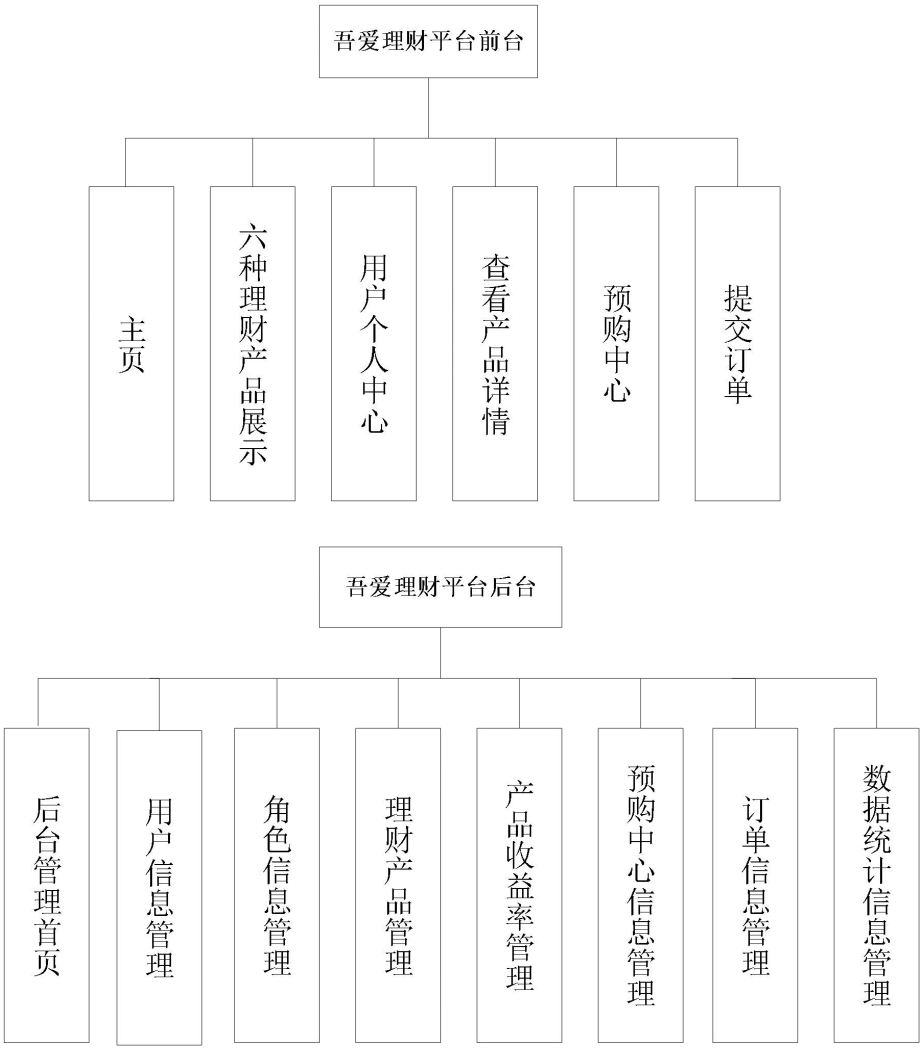

图 3-1 整体功能结构图

3.1.1 E-R 图设计

在对网站进行需求和设计分析之后，总结出本“吾爱理财平台”所需的各大实体对象，其中包含普通用户、管理员、管理员角色类型、理财产品、产品收益率、预购中心、订单等。本“吾爱理财平台”的 E-R 图，具体见图 3-2。

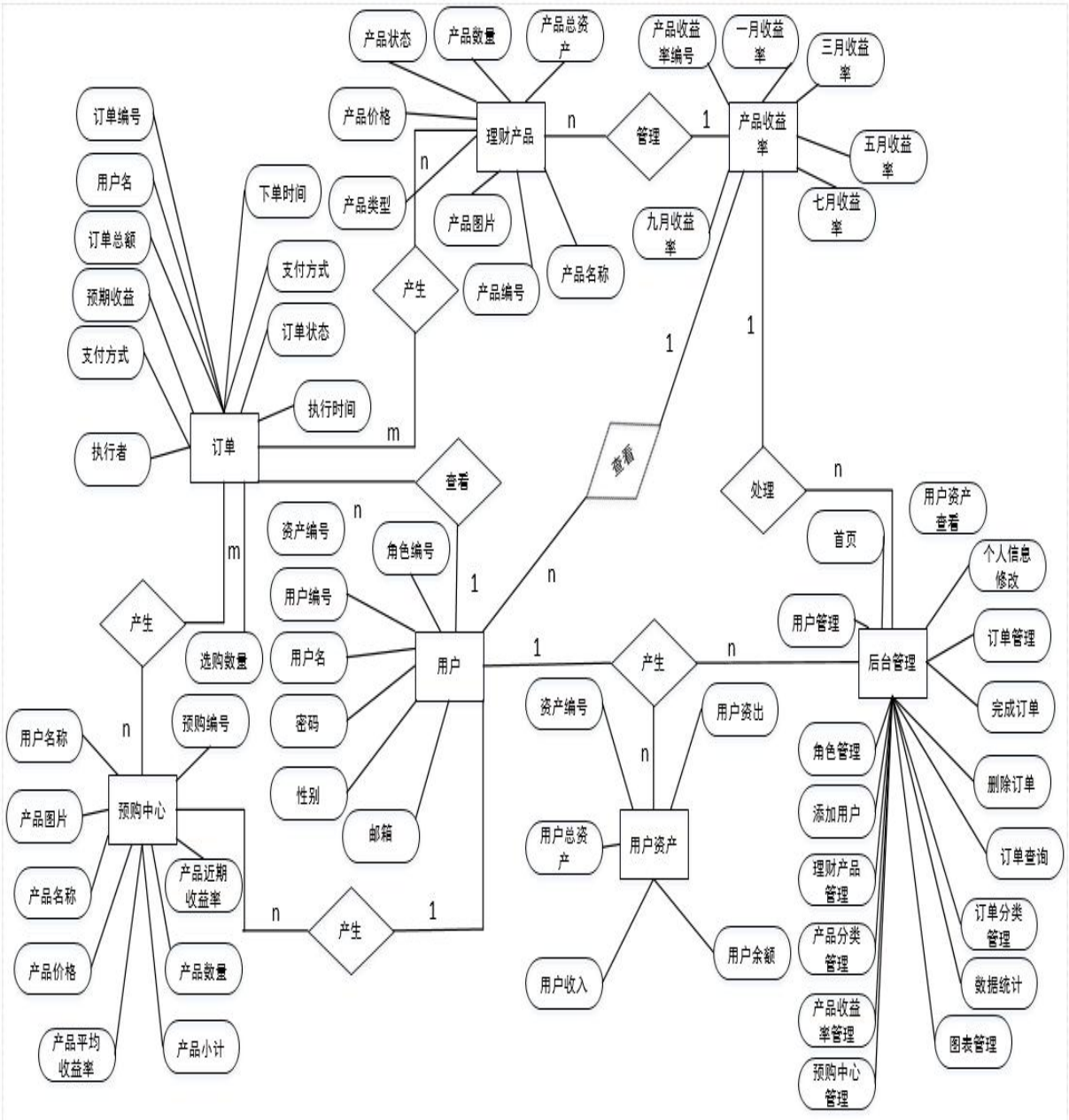

图 3-2 E-R 图

3.1.2 网站用例图设计

通过对本吾爱理财平台的需求分析，根据前后端的使用者来分类，确定了本吾爱理财平台的两个重要角色，即前端用户角色和管理员角色。

(1)用户角色的用例图主要是说明了前端用户可进行的操作：浏览前端所有界面；

查看个人信息；浏览理财产品；查看产品细节；添加至预购中心；直接结算某一支产品；形成订单；提交订单；进行支付等。具体信息见图 3-3。

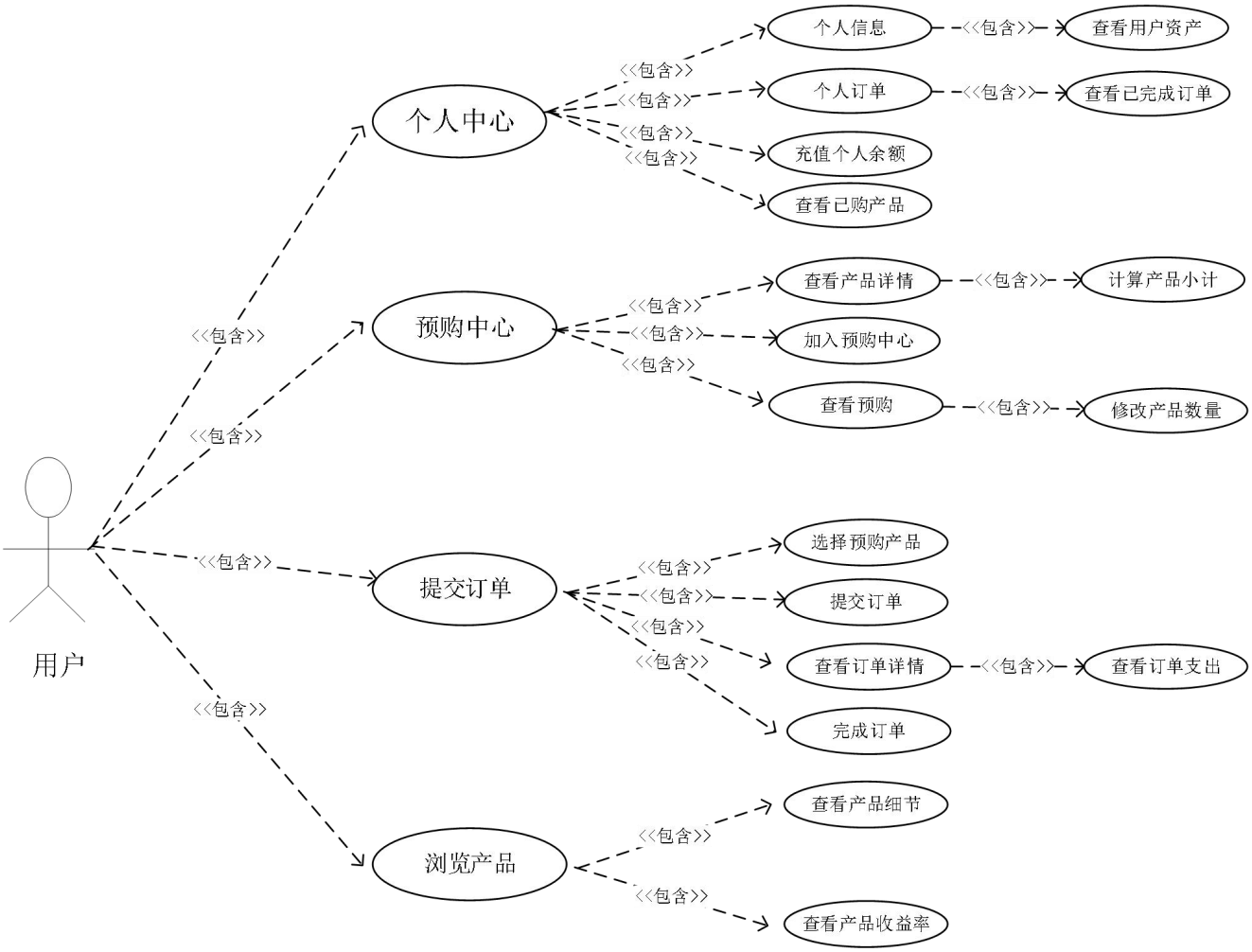

图 3-3 用户用例图

(2)管理员角色用例图主要是说明管理员可以进行的操作，这包含了对用户信息、角色信息管理、管理理财产品信息、理财分类信息管理、预购中心信息、订单管理、订单状态管理、数据图表信息管理、后台首页信息的管理，具体信息见图 3-4。

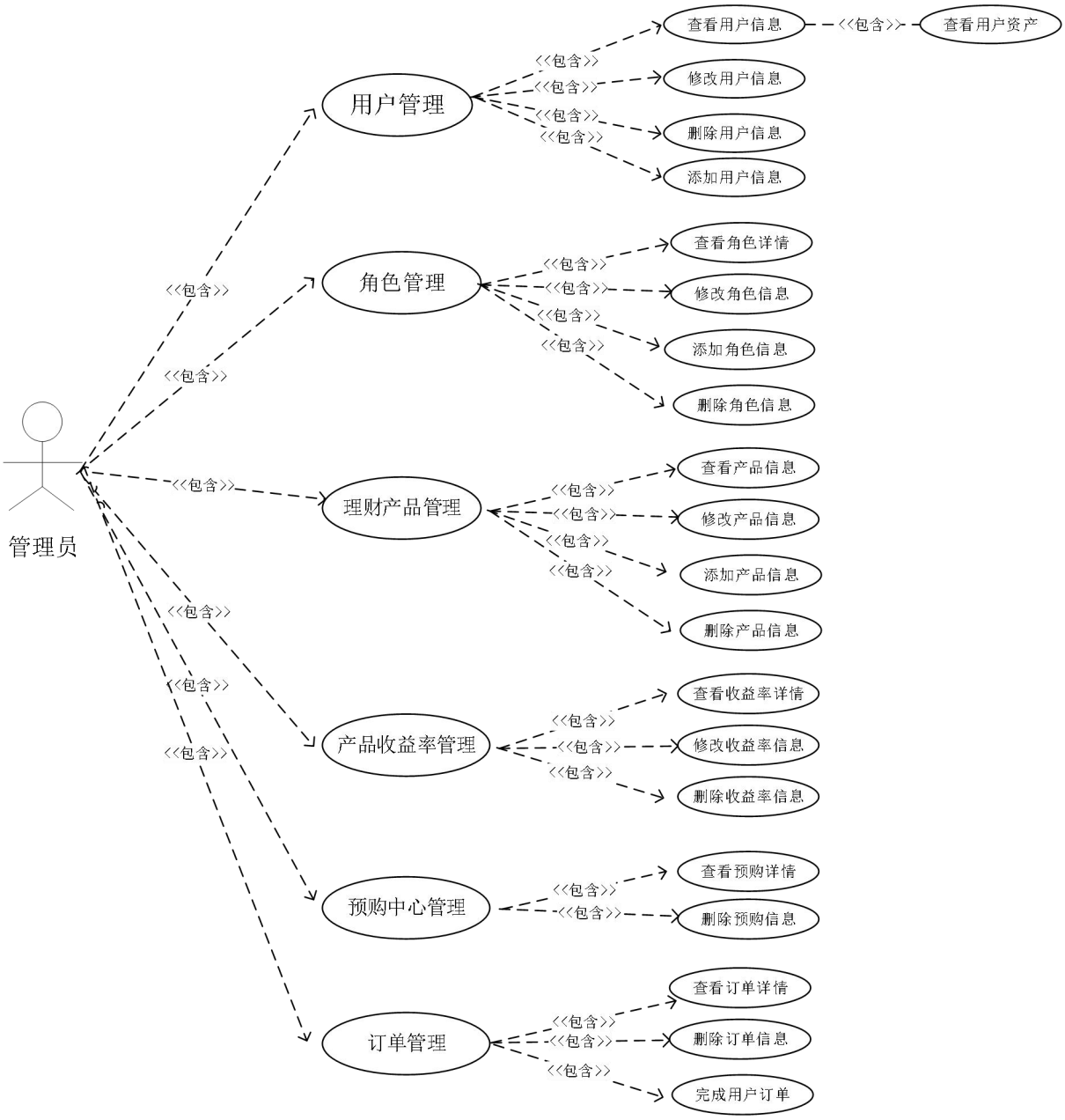

图 3-4 管理员用例图

3.1.3 网站类图设计

本吾爱理财平台的类图大致上描述了用户、理财产品、预购中心、订单等实体类的结构。

(1)用户类图展示了它所构建的用户实体、实体内部结构间关系，定义了用户实体类、用户的角色，用户的状态，用户的资产中各个参数的参数类型及信息。为保证数据的安全，将数据定义为 Private，调用时用 get()、set()方法来调用见图 3-5。

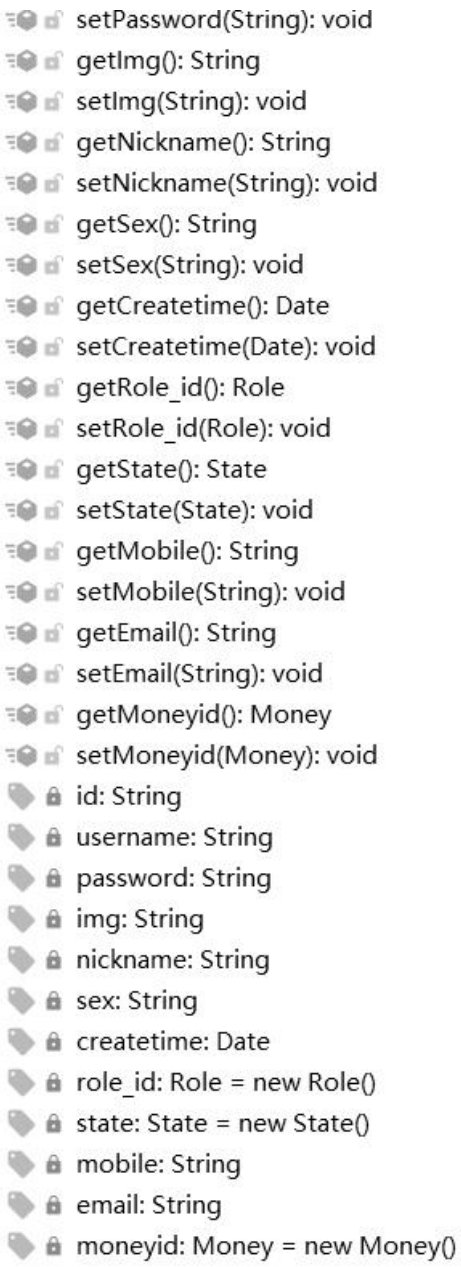

图 3-5 用户类图

(2) 用户资产类图是显示了用户资产模块的静态结构，特别是用户资产类、该用户资产类的内部结构，通过该类图可以知道该类有 5 个参数及其类型、调用方法，详情见图 3-6。

图 3-6 用户资产类图

(3) 用户角色类图展示了它所构建的用户角色实体、实体内部结构间关系，定义了用户角色实体类、用户的编号，角色的名称，用户的描述中各个参数的参数类型及信息。为保证数据的安全，将数据定义为 Private，调用时用 get()、set()方法来调用见图 3-5。

图 3-7 用户角色类图

(4) 用户状态类图是显示了用户状态的静态结构，特别是用户状态编号类、该用户状态类的内部结构，通过该类图可以知道该类有 3 个参数及其类型、调用方法，为保证数据的安全，将数据定义为 Private，详情见图 3-8。

图 3-8 用户状态类图

(5)理财产品类图描述了管理员实体及其实体内部结构间的关系，定义了理财产品实体类、产品的经理，产品的最近收益率，产品的状态、产品的类别中各个参数的参数类型及信息。具体见图 3-9

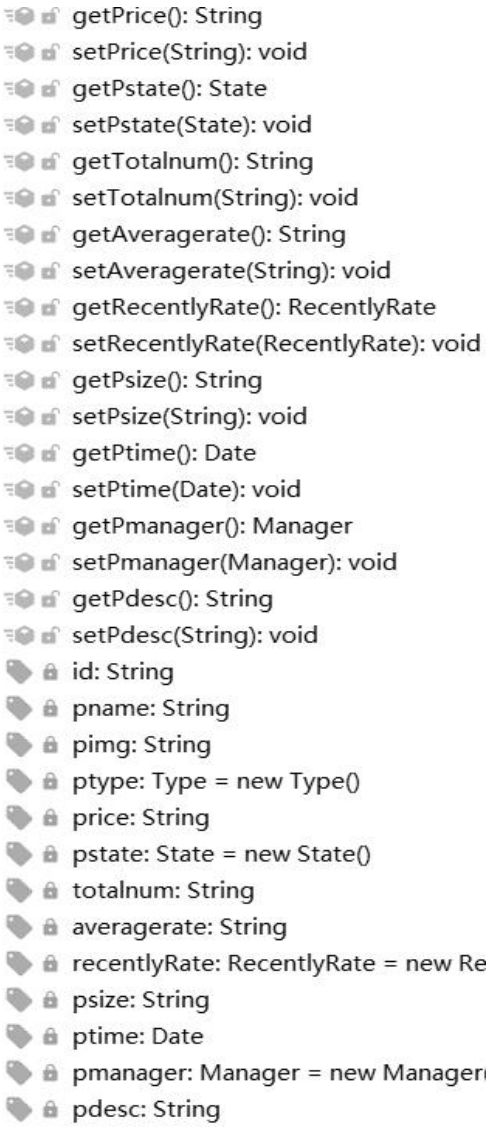

图 3-9 理财产品类图

(6)产品经理类图展示了产品经理实体及其内部 3 个参数之间的关系结构，产品经理的信息数据的获取也是通过 get()、set()方法调用的，为保证数据的安全，将数据定义为 Private 详情见图 3-10。

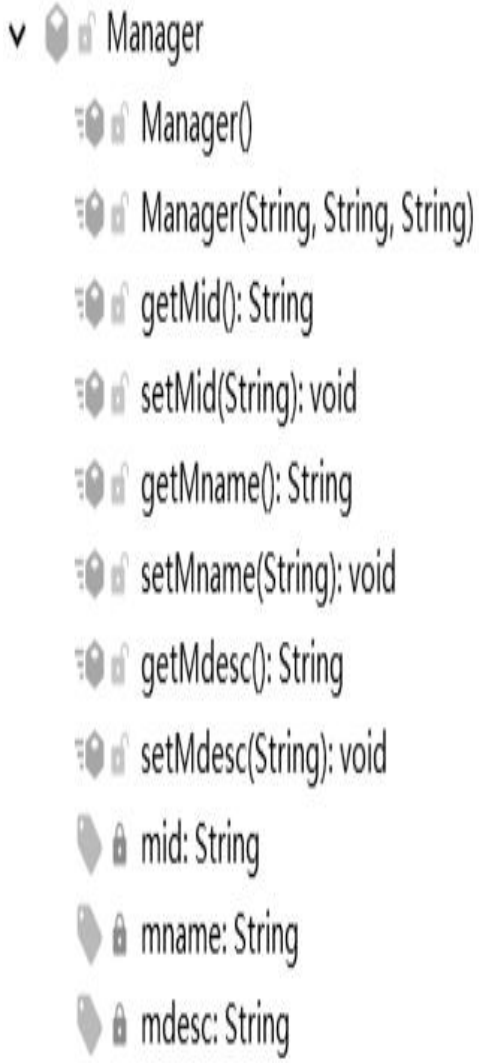

图 3-10 产品经理类图

(7) 产品收益率类图展示了产品收益率实体及其内部结构的 8 个参数。为了实现查看每支产品的收益率，将参数设为私有 Private，并调用 set()、get()方法来获取数据，详情见图 3-11。

图 3-11 产品收益率类图

(8)产品状态类图展示了产品状态类和其内部的 3 个参数之间的关系结构，这些参数的调用使用 get()、set()方法，特别是产品状态编号类、该产品状态类的内部结构为保证数据的安全，将数据定义为 Private 详情见图 3-12。

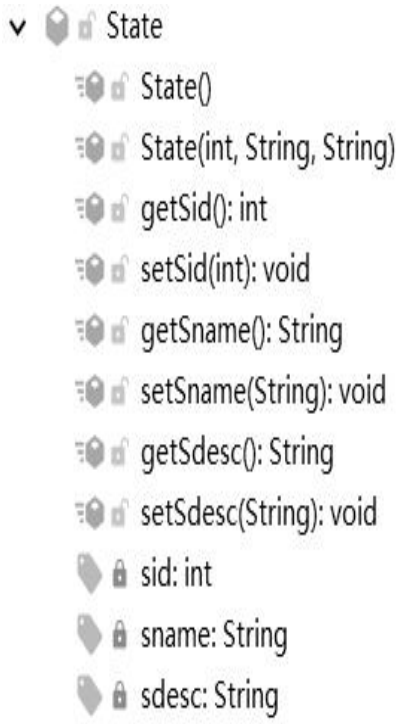

图 3-12 产品状态类图

(9)产品类型类图是显示了产品类型的静态结构，特别是产品类型编号类、该用户状态类的内部结构，通过该类图可以知道该类有 3 个参数及其类型、调用方法，为保证数据的安全，将数据定义为 Private，详情见图 3-13。

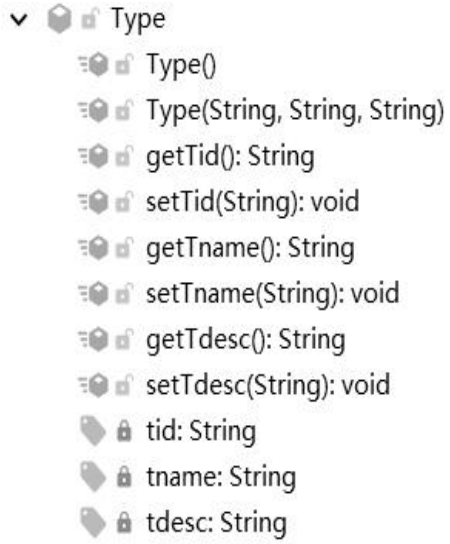

图 3-13 产品类型类图

(10) 预购中心类图描述了预购中心实体，与产品类和用户类产生了紧密联系，如 username,pname 字段，通过该类图可以知道该类有 7 个参数及其类型、调用方法为保证数据的安全，将数据定义为 Private 具体见图 3-14。

图 3-14 预购中心类图

(11)订单类图描述了订单实体及其实体内部结构间的关系，定义了订单实体类、订单的状态，订单的详情，订单的类型中各个参数的参数类型及信息。为保证数据的安全，将数据定义为 Private 具体见图 3-15

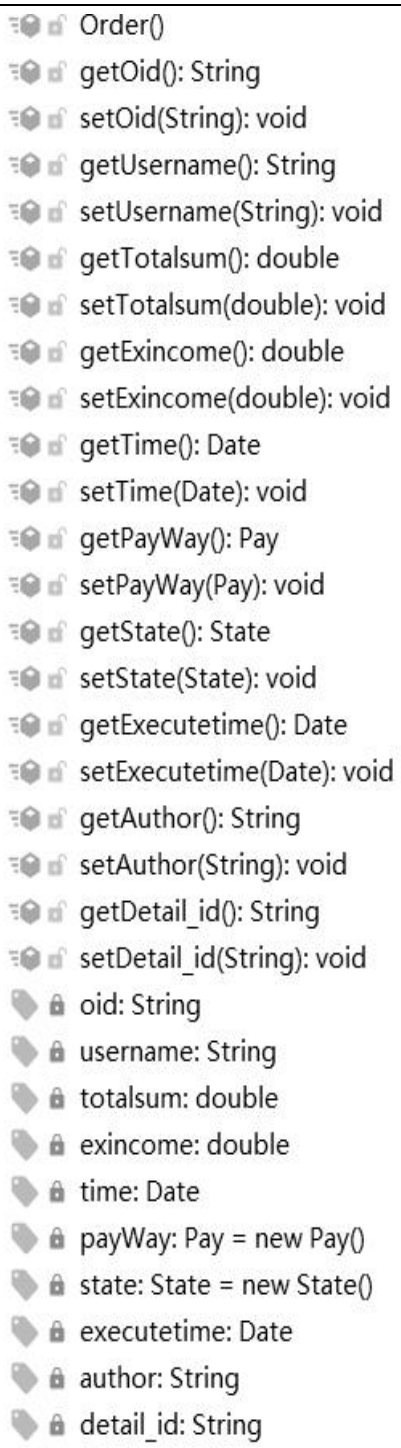

图 3-15 订单类图

(12)订单详情类图展示了订单详情类和其内部的 6 个参数之间的关系结构，这些参数的调用使用 get()、set()方法，为保证数据的安全，将数据定义为 Private，订单的详情中各个参数的参数类型及信息，详情见图 3-16。

图 3-16 订单详情类图

(13)订单支付方式类图是显示了订单支付方式的静态结构，特别是支付的编号和描述的字段，通过该类图可以知道该类有 3 个参数及其类型、调用方法，为保证数据的安全，将数据定义为 Private 详情见图 3-17。

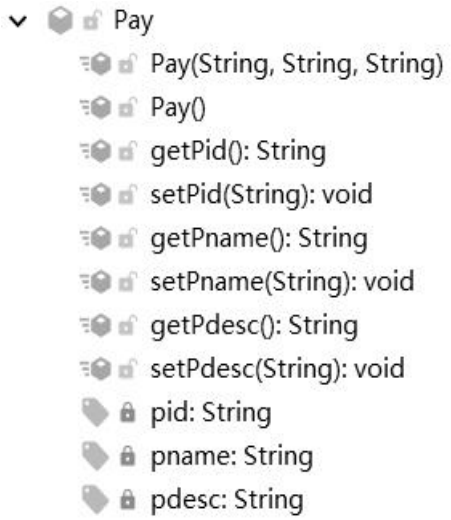

图 3-17 订单支付方式类图

(14)订单状态类图展示了订单状态实体及其内部结构的 3 个参数。为了保证用户提交的寄养订单是数据的安全性，将参数设为私有 Private，并调用 set()、get()方法来获取数据，详情见图 3-18。

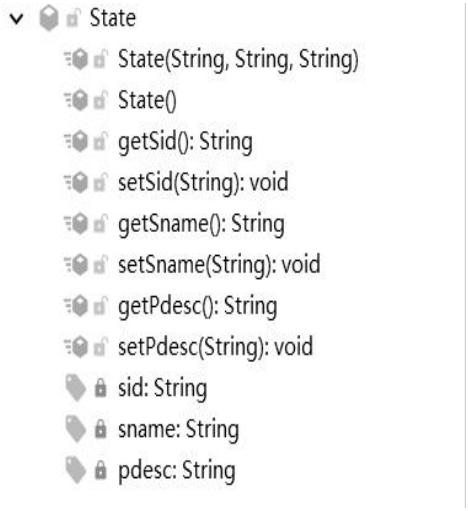

图 3-18 订单状态类图

3.2 网站数据库设计

(1)本吾爱理财平台设计的用户表用来保存使用本平台的所有用户包括管理员的基本信息，用户注册需要输入用户名、密码及邮箱进行注册，用户的用户名是不可重复的。数据将通过添加方法等保存在 g_d_user 表中，登录时输入用户名及密码，与数据库对比一致方可登录成功，普通用户在登录成功直接跳转到平台首页，若是管理员则进入管理员登录页面，下图为 g_d_user 表，该用户表详情见表 3-1。

表 3-1 用户表(g_d_user)

字段	类型	备注	约束
id	varchar2(20)	用户编号	主键，自增
username	varchar2(50)	用户名	not null,unique
password	varchar2(50)	登录密码	default(null)
img	varchar2(20)	用户头像	default(null)
nickname	varchar2(50)	用户昵称	default(null)
sex	varchar2(5)	用户性别	default(null)
createtime	date	创建用户时间	default(null)
role_id	varchar2(10)	用户角色 id	default(null)
state	number(10)	用户状态	default(null)
mobile	varchar2(50)	用户联系方式	default(null)
email	varchar2(50)	用户邮箱	default(null)
moneyid	varchar2(20)	用户资产 id	default(null)

(2)本吾爱理财平台设计的角色管理表，保存使用本平台所有用户角色信息，管理员可根据角色信息获取不同角色的权限，进行权限功能的实现。不同管理员角色的权限功能不同具体详情见表 3-2。

表 3-2 角色管理表(g_d_u_role)

字段	类型	备注	约束
rid	varchar2(20)	角色编号	主键，自增
rname	varchar2(50)	角色名称	unique
rdesc	varchar2(50)	角色描述	default(null)
rdate	date	角色创建/修改日期	default(null)
rauthor	varchar2(10)	角色创建/修改者	default(null)

(3)本吾爱理财平台的用户状态管理表，保存用户状态，状态管理表共有 3 个字段，可通过修改用户的状态来控制用户是否能够登录或者消费，管理员也可在后台管理中修改用户状态的描述。具体字段信息见表 3-3。

表 3-3 状态管理表(g_d_u_state)

字段	类型	备注	约束
sid	number(10)	用户状态 id	主键，自增
sname	varchar2(30)	用户状态名称	default(null)
sdesc	varchar2(50)	用户状态描述	default(null)

(4)本吾爱理财平台的用户资金账号表，用于保存用户当申请开通账户时的信息，只有当用户点击开通账户以后才可以去购入理财产品。该用户资产账号表共有 6 个字段，普通用户也可在个人中心查看细节，详情见表 3-4。

表 3-4 用户资金账号表(g_d_u_money)

字段	类型	备注	约束
mid	varchar2(20)	用户资金账号 id	主键，自增
totalmoney	number(20,2)	用户总资产	default(null)
income	number(20,2)	用户收入	default(null)
pay	number(20,2)	用户支出	default(null)
balance	number(20,2)	用户余额	default(null)

(5)本吾爱理财平台的理财产品表，前台展示供用户选购，后台供管理员管理理财产品的信息，包括对理财产品的删除与编辑。该表共有 13 个字段，且每个字段之间互相有联系详情见表 3-5。

表 3-5 理财产品表(g_d_product)

字段	类型	备注	约束
id	varchar2(20)	产品编号	主键，自增
pname	varchar2(50)	产品名称	unique
pimg	varchar2(10)	产品图片	unique
pmanager	varchar2(20)	产品经理	default(null)
pdesc	varchar2(50)	产品描述	default(null)

续表 3-5

字段	类型	备注	约束
ptype	varchar2(10)	产品类型	default(null)
price	varchar2(30)	产品价格	default(null)
pstate	varchar2(10)	产品状态	default(null)
totalnum	number(10)	产品数量	default(null)
averagerate	varchar2(30)	产品的平均收益率	default(null)
recentlyrate	varchar2(10)	产品近期收益率	default(null)
psize	varchar2(10)	产品总资产	default(null)
ptime	date	产品上市时间	default(null)

(6)本吾爱理财平台所设计的产品类型表使用来区分不同的理财产品，前台可供用户分类型去选购产品，后台管理可供管理员分类去查看和管理理财产品，该产品类型表的字段详情信息见表 3-6。

表 3-6 产品类型表(g_d_p_type)

字段	类型	备注	约束
tid	varchar2(10)	产品类型 id	主键，自增
tname	varchar2(30)	产品类型名称	default(null)
tdesc	varchar2(50)	产品类型描述	default(null)

(7)本吾爱理财平台的产品状态表，保存用户提交志愿者申报的信息，用户在前端页面上填写志愿者申请表，点击提交按钮，通过添加方法等，志愿者信息会上传到数据库表中。志愿者表的字段共有 9 个，详情见表 3-7。

表 3-7 产品状态表(g_d_p_state)

字段	类型	备注	约束
sid	number(10)	产品状态 id	主键，自增
sname	varchar2(20)	产品状态名称	default(null)
sdesc	varchar2(20)	产品状态描述	default(null)

(8)本吾爱理财平台的产品经理表，保存的是所有产品的产品经理，以及产品经理的业绩描述。在产品详情中可以了解到该产品的所属产品经理，且每个产品经理的编号都是唯一的。该表有 3 个字段，详情见表 3-8。

表 3-8 产品经理表(g_d_p_manager)

字段	类型	备注	约束
mid	varchar2(20)	产品经理编号	主键，自增
mname	varchar2(20)	产品经理名称	default(null)
mdesc	varchar2(50)	产品经理描述	default(null)

(9)本吾爱理财平台所设计的产品近期收益率表，该表用于保存产品发布后的近期收益率，供管理员给用户计算消费金额。该表共有 7 个字段，且每个产品经理的编号都是唯一的。字段详情信息见表 3-9。

表 3-9 产品近期收益率表(g_d_p_recentlyrate)

字段	类型	备注	约束
rid	varchar2(10)	产品收益率编号	主键，自增
jan	varchar2(10)	一月收益率	default(null)
mar	varchar2(10)	三月收益率	default(null)
may	varchar2(10)	五月收益率	default(null)
july	varchar2(10)	七月收益率	default(null)
sep	varchar2(10)	九月收益率	default(null)
nov	varchar2(10)	十一月收益率	default(null)

(10)本吾爱理财平台的用户预购中心表，用来保存所有用户的所有预购产品记录，用户可对预购中心的产品预购数量进行增加，实时的总额也将显示供用户自主查看。管理员在后台也可以根据用户名对预购产品进行分类与分析，该用户预购中心表共有 8 个字段，详情见表 3-10。

表 3-10 用户预购中心表(g_d_u_cart)

字段	类型	备注	约束
cid	varchar2(20)	预购编号	主键，自增
username	varchar2(50)	用户名称	unique
pimg	varchar2(10)	产品图片	default(null)
pname	varchar2(50)	产品名称	default(null)
price	varchar2(30)	产品价格	default(null)
averagerate	varchar2(10)	产品平均收益率	default(null)
recentlyrate	varchar2(10)	产品近期收益率	default(null)
pnum	number(20)	产品数量	default(null)
sum	varchar2(20)	产品小计	default(null)

(11)本吾爱理财平台的订单信息表，用来保存用户提交的所有的订单信息，普通用户可以在个人中心查看到自己已完成的订单和订单详情。后台管理员可以对订单进行批量的删除，记录操作的管理员和执行时间，见表 3-11。

表 3-11 订单信息表(g_d_order)

字段	类型	备注	约束
oid	varchar2(20)	订单编号	主键，自增
username	varchar2(30)	用户名	unique
totalsum	number(10,2)	订单总额	default(null)
Exincome	number(10,2)	预期收益	default(null)
Author	varchar2(20)	执行者	default(null)

(12)本吾爱理财平台的订单支付方式表，用来保存用户提交订单时候选择的支付方式，其中订单支付编号设置了唯一性，用户在支付时可以选择支付的方式，以及管理员可以对支付方式进行描述介绍，详情见表 3-11。

表 3-12 订单支付方式表(g_d_o_pay)

字段	类型	备注	约束
pid	varchar2(10)	订单支付方式编号	主键，自增
pname	varchar2(20)	支付名称	default(null)
pdesc	varchar2(50)	支付描述	default(null)

(13)本吾爱理财平台的订单状态表，用来保存用户提交的订单的状态，包括等待中，已支付，已取消，已完成四个状态，前台普通用户可以查看到已完成的所有订单，后台管理员可以对所有订单状态类别进行分类管理，见表 3-13。

表 3-13 订单状态表(g_d_o_state)

字段	类型	备注	约束
sid	varchar2(10)	订单状态编号	主键，自增
sname	varchar2(20)	状态名称	default(null)
pdesc	varchar2(50)	状态描述	default(null)

(14)本吾爱理财平台的订单详情表，用来保存用户提交的所有的订单信息详情，普通用户可以在个人中心查看到自己已完成的订单详情。后台管理员可以对每条订单进行查看，了解每条订单的细节，见表 3-14。

表 3-14 订单详情表(g_d_o_detail)

字段	类型	备注	约束
id	varchar2(20)	订单详情编号	主键，自增
pname	varchar2(20)	产品名称	unique
pimg	varchar2(20)	产品图片	default(null)
price	varchar2(20)	产品单价	default(null)
averagerate	varchar2(2，0)	产品平均收益率	default(null)
username	varchar2(20)	购买者	unique

3.3 本章小结

本章节主要是对本平台的功能设计和数据设计做了大致的描述，先是使用功能结构图指明本网站有哪些功能模块；而后通过 E-R 图来表述本吾爱理财平台的几个实体之间的关系；用例图是概括了前端用户和后端管理员的操作权限；最后通过类图的方式将本网站的用户信息、角色信息、用户状态信息、理财产品信息、产品收益率信息、预估中心信息、订单信息，订单详情信息的具体结构展示出来。

4 本吾爱理财平台的详细设计与实现

4.1 用户模块的设计与实现

4.1.1 用户注册功能的实现

(1)算法设计思路

游客需注册账号方能成为本网站的用户，这里对用户名密码注册方式进行详细说明。其一是用户名、密码、邮箱注册，用户输入用户名、密码、邮箱，单击提交即可完成注册，并且在当注册的同时给该用户添加一个用户资产账号。本网站对用户名设置限制，即用户名不能重复，前端通过 Axios 将用户名作为参数传递给后台，后台 Controller 通过 Service 层检查用户名是否重复，Service 层再调用 Mapper 层通过 SQL 语句来查询数据库，若返回值存在，则该用户名已被注册。此处还对用户名、密码输入框做了焦点定位，若输入框为空，会弹出友好提示框告诉用户，该内容不能为空。用户点击注册后，用户名和密码这两个参数传递给业务层，业务层调用持久层方法，在 SQL 语句中，将两个参数赋值到语句中。

(2)实现代码

```
@ApiOperation("注册的接口")
@GetMapping(value = "/regist/{username}/{password}/{email}")
Public boolean regist(@ApiParam("用户名") @PathVariable("username") String username,
@ApiParam("密码") @PathVariable("password") String password,
@ApiParam("邮箱") @PathVariable("email") String email){
    User user = new User(username, password,email);
    userBiz.addMoney();
    return userBiz.regist(user);
}
<script>
register: function () {
this.$confirm("确定注册?", "提示", {
confirmButtonText: "确定",
cancelButtonText: "取消",
type: "info",
```

```
})
.then(() => {
let username = this.ruleForm.username;
let password = this.ruleForm.password;
let email = this.ruleForm.email;
this.$http
.get(
"http://localhost:8081/user/regist/" +
username +
"/" +
password +
"/" +
email,
{ xhrFields: { withCredentials: true } },
{ crossDomain: true }
)
.then((res) => {
console.log(res);
if (res.data) {
this.$message({
showClose: true,
message: "恭喜你注册成功！请点击去登录",
type: "success",
});
}
setTimeout(()=>{
this.reload();
},1000)
})
.catch(() => {
this.$message({
type: "error",
message: "注册失败，该用户已注册！",
});
```

```
});
})
.catch(() => {
this.$message({
type: "info",
message: "已取消注册",
});
});
},
<script>
```

4.1.2 用户登录功能的实现

(1)算法设计思路

由于本网站对用户的用户名设有唯一限制，用户登录时只需正确填写用户名和密码。此处写了表单格式验证功能，若信息输入无误，点击登录按钮，将用户名和密码通过 Axios 异步传输到后台控制层，Controller 调用业务层的登录方法，业务层调用持久层的登录 SQL 语句，Controller 根据是否返回数据来判断用户名和密码是否正确，将该用户的信息放进作用域中。

(2)实现代码

```
@ApiOperation("登录的接口")
@GetMapping(value="/login/{username}/{password}")
public List<HashMap<String, Object>> login(@ApiParam("用户名") @PathVariable("username")
String username, @ApiParam("密码") @PathVariable("password") String password,
HttpServletRequest request){
    List<HashMap<String, Object>> list = userBiz.login(username, password);
    if (list != null) {
        String userName = (String) list.get(0).get("USERNAME");
        request.getSession().setAttribute("username", userName);
    }
    return list;
}
<script>
  login: function () {
        this.dialogbuttonVisible = true;
          let username = this.tabelData.username;
```

```
            let password = this.tabelData.password;
            this.$http.get("http://localhost:8081/user/login/" +username +"/" +password,
                { xhrFields: { withCredentials: true } },
                { crossDomain: true }
              )
              .then((res) => {
                console.log(res);
                if (res.data!=null) {
                  this.$store.commit("uInfo/setUserInfo", res.data[0]);
                  localStorage.setItem("loginData", JSON.stringify(res.data[0]));
                  this.ruleForm.role_id = res.data[0].ROLE_ID;
                  this.ruleForm.state = res.data[0].STATE;
                  //this.ruleForm.error = res.data.status;
                  this.msg=true;
                }
              })
  }
  </script>
```

4.1.3 用户对个人信息操作功能

(1)算法设计思路

用户登录账号后，在个人中心板块中点击信息修改，先获取当前用户的 Id，将用户 Id 通过 Axios 传递给 Controller，Controller 调用 Service 业务层方法，业务层再调用 Mapper 持久层进行数据库信息查询，根据返回的信息，将其赋予前端定义的集合中，通过调用该集合回显当前用户信息。用户点击信息修改，会有模态框弹出，用户在此将修改自己想要修改的信息，点击提交即可，将修改后的数据传输到后台完成修改。此时，前端向后台发送一次访问请求，调用修改的方法，完成修改信息的操作。

(2)实现代码

```
  @ApiOperation("用户信息接口")
      @GetMapping(value="/findUser/{index}")
      public Map<String, Object> findUser(@ApiParam("分页的参数") @PathVariable("index")
  Integer index){
          if (index==null){
              index=1;
          }
```

```
            int size=5;
            List<User> list = userBiz.findUser(index, size);
            int count = userBiz.countUser();
            Map<String, Object> map = new HashMap<String, Object>();
            map.put("list", list);
            map.put("index", index);
            map.put("count", count);
            return map;
}
<script>
submitForm(formName) {
        this.$refs[formName].validate((valid) => {
          if (valid) {
            this.$confirm("是否修改？ ", "tips", {
              confirmButtonText: "OK",
              cancelButtonText: "Cancel",
              type: "warning",
            })
              .then(() => {
                let username = this.userData.username;
                let password = this.userData.password;
                let img = this.userData.img;
                let nickname = this.userData.nickname;
                let sex = this.userData.sex;
                let mobile = this.userData.mobile;
                let email = this.userData.email;
                this.$http .get("http://localhost:8081/user/updateUserByUsername/" +
username +"/" +password +"/" +img + "/" +nickname +"/" +sex + "/" +mobile +"/" +email,
                    { xhrFields: { withCredentials: true } },{ crossDomain: true }
                  )
                  .then((req) => {
                    if (req.data != null) {
                      this.dialogFormEVisible = false;
                      this.$message({
```

```
                        showClose: true,
                        message: "修改成功",
                        type: "success",
                    });
                    this.reload();
                }
            });
        })
        .catch(() => {
            this.$message({
                type: "info",
                message: "已取消修改!",
            });
        });
    } else {
        return false;
    }
});
},
</script>
```

4.1.4 用户退出登录功能

(1)算法设计思路

用户在个人中心板块，可以选择退出当前用户，那么平台会将存储的 Vuex 和 localStorage 的信息给移除，前端将用户 username 作为参数，用 Axios 传递到后台，在用户的控制器中调用根据用户 username 清除后端 session，前端通过先清除本地 localStorage 中的信息，再清除 Vuex[13]中的用户信息和用户资产信息实现业务功能。

(2)实现代码

```
@ApiOperation("登出的接口")
@GetMapping(value = "/loginOut")
        public boolean loginOut(HttpSession session){
    boolean b = false;
    //清除 session  中的用户
    session.removeAttribute("username");
    if(session.getAttribute("username")==null){
```

```
                b=true;
            }
            return    b;
    }
        <script>
        handleCommand(command) {
            this.$confirm("退出当前用户，是否退出?", "提示", {
              confirmButtonText: "确定",
              cancelButtonText: "取消",
              type: "warning",
            })
              .then(() => {
                this.$http.get(
                    "http://localhost:8081/user/loginOut",
                    { xhrFields: { withCredentials: true } },{ crossDomain: true }
                )
                .then((res) => {
                  console.log(res);
                  localStorage.removeItem("loginData");
                  this.$store.commit("setUserInfo", {});
                  localStorage.removeItem("money");
                  this.$store.commit("setMoneyInfo", {});
                  if (res.data == true) {
                    this.$message({
                      type: "success",
                      message: "退出成功!",
                    });
                    this.$router.replace({ path: "/frontLogin" });
                  }
                });
              })
              .catch(() => {
                this.$message({
                  type: "info",
```

```
                    message: "已取消操作!",
                });
            });
        }
    },
</script>
```

4.1.5 用户充值个人余额功能

(1)算法设计思路

用户在完成注册登录后，可以跳转到其个人中心板块的个人详情页面，用户可以查看左侧饼状图了解目前的个人账户余额状况，如果想要充值余额，可以点击上方绿色充值按钮进行模拟支付功能。

(2) 实现代码

```
<script>
  recharge: function () {
        this.$prompt("请输入充值金额", "金额数量确认", {
          confirmButtonText: "确定",
          cancelButtonText: "取消",
          inputValue: "100",
          center: "true",
        })
          .then(({ value }) => {
            if (value == null) {
              value = 100;
            }
            this.$httpget("http://localhost:8081/user/updateMoney/" +
                  this.moneyid +"/" +value
              )
              .then((res) => {
                console.log(res.data);
                if (res.data != null) {
                  this.$notify({
                    title: "成功",
                    message: "请前往支付~",
                    type: "success",
```

```
                    });
                }
                window.location.href = "/pay" + value;
            });
        })
        .catch(() => {
          this.$message({
            type: "info",
            message: "取消充值！",
          });
        });
    },
</script>
```

4.1.6 用户查看订单详情功能

(1)算法设计思路

对于用户成功提交的订单信息，用户可以在个人中心的个人详情板块查看到。用户提交的订单信息是通过前端将用户 username 作为参数，用 Axios 传递到后台，在订单模块的控制器中调用根据用户 username 回显订单信息，回显的信息以 JSON 格式传递到前端，前端通过 List 赋值调用该用户的订单信息。

(2)实现代码

```
@ApiOperation("根据当前用户名查询订单接口")
    @GetMapping(value="/findOrderByUsername/{index}")
    public   Map<String, Object> findOrderByUsername(@ApiParam("分页的参数")
@PathVariable("index") Integer index, HttpServletRequest request) {
        if(index==null){
            index=1;
        }
        int size=5;
         Map<String, Object> map = new HashMap<String, Object>();
         String username=(String) request.getSession().getAttribute("username");
         int count =orderBiz.countOrderByUsername(username);
         List<Order> list= orderBiz.findOrderByUsername(index,size,username);
         map.put("list",list);
         map.put("index",index);
```

```
        map.put("count",count);
        return map;
    }
```

4.2 理财产品模块的设计与实现

4.2.1 理财产品陈列展示模块

(1)算法设计思路

在本网站前端的任一界面的导航栏中都能够看到理财产品板块，包括“股票”、“基金”、“债券”、“银行储蓄”、“保险”、“黄金”六大板块，选择想要了解的一种进入该页面，该页面陈列展示了某一理财种类的所有的可挑选产品。提前将理财产品的图片存到服务器上，当前端实现理财产品信息展示，将当前理财产品类型作为参数，通过 Axios 向后台发送一次请求，获取当前类型的理财产品信息，理财产品管理模块的 Controller 控制层调用 Service 业务层里按类型查找理财产品信息的方法，Service 业务层调用 Mapper 持久层的方法来实现与数据库信息的查询，返回的数据以 List 集合方式存储在控制器，将该信息以 JSON 格式传送到前端，定义一个新的 List，原 List 将该信息赋值给新 List，通过调用新建的 List 获取数据。

(2)实现代码

```
 @ApiOperation("普通用户根据类型查询产品")
      @GetMapping(value = "/CustomerfindProductByType/{ptype}")
    public Map<String, Object> CustomerfindProductByType(@PathVariable("ptype") String
 ptype){
        List<Product> list = productBiz.CustomerfindProductByType(ptype);
        int count = productBiz.countProductByType(ptype);
        Map<String, Object> map = new HashMap<String, Object>();
        map.put("list", list);
        map.put("ptype", ptype);
        map.put("count", count);
        return map;
    }
```

4.2.2 理财产品详情信息展示

(1)算法设计思路

在理财产品展示界面，用户随机点击一个产品的了解详情按钮，即可跳转到该理财

产品的详情界面，详情界面向用户展示了该理财产品的详细信息，如左侧产品类型介绍视频、名称、产品经理、产品收益率、产品总资产等等，在点击产品了解详情的同时，将产品的 Id 放在 URL 里，一起传输到该产品详情界面，单独打开产品详情界面，则无参数，无法进入。在产品详情界面，通过分解 URL，获得该产品的 Id，根据获得的产品 Id 向后台申请查询该产品的信息。

(2)实现代码

```
@ApiOperation("普通用户根据 id 查询产品")
@GetMapping(value = "/CustomerfindProductById/{id}")
public Map<String, Object> CustomerfindProductById(@PathVariable("id") String id) {
    List<Product> list = productBiz.CustomerfindProductById(id);
    //System.out.println(list.get(0).getPname());
    Map<String, Object> map = new HashMap<String, Object>();
    map.put("list", list);
    return map;
}
```

4.3 订单模块的设计与实现

4.3.1 提交订单申请

(1)算法设计思路

用户确定想要购入产品时，有两种方式提交订单。其一通过直接在该理财产品详情页结算该理财产品输入购入数量进入订单确认最后去支付的方式，其二通过批量的形式，用户先将意向的产品添加至个人预购中心中，最后进行批量结算支付的方式完成。第二种方式的实现思路是通过 ElementUI 框架的 el-table 中多选框 handleSelectionChange 属性去获取选择的产品信息，然后将获取到的产品信息存入到 data 中定义好的数组中存储。并将这个数组转换成 Json 字符串提交到后端，后端再将字符串转为 Json 数组的形式传递至 Mapper 中。

(2)实现代码

```
@RequestMapping(value="/addOrder")
@ResponseBody
public boolean addOrder(String detail,double totalSum,String payWay,String detail_id ,String
username) throws ParseException {
    Date randomDate = randomDate("2020-01-01 ", "2022-11-30 ");
    java.sql.Date time=new java.sql.Date(randomDate.getTime());
```

```
            JSONArray detail1 =JSON.parseArray(detail);
            boolean flag=cartBiz.insertAll(detail_id,detail1,username);
            System.out.println(flag);
            double exincome= totalSum*(cartBiz.avg(detail_id)/100);
          return    cartBiz.addOrder(username,totalSum,exincome,time,payWay,detail_id);
}
    <script>
    pay: function () {
        this.$confirm("信息是否正确，是否继续?", "提示", {
            confirmButtonText: "确定",
            cancelButtonText: "取消",
            type: "warning",
        })
            .then(() => {
                this.balance = this.$store.getters["uInfo/getUserMoneyInfo"].balance;
                if(this.balance > this.totalsum){
                    let detail_id = Math.round(Math.random() * 1000);
                    let detail = JSON.stringify(this.tableData);
                    var that = this;
                    this.$jquery.ajax({
                        url: "http://localhost:8081/cart/addOrder",
                        data:
"detail="+detail+"&totalSum="+this.totalsum+"&payWay="+this.payWay+"&detail_id="+detail_id
+"&username="+this.username,
                        type: "post",
                        dataType: "json",
                        success: function (res) {
                            if (res) {
                                that.$notify({
                                    title: "成功",
                                    message: "请支付",
                                    type: "success",
                                });
                                    window.location.href = "/pay" + that.totalsum.toFixed(2);
```

```
                    }
                  },
                });
              }else {
                this.$notify.error({
                  title: "错误",
                  message: "用户余额不足，无法支付结算！",
                });
              }
            })
            .catch(() => {
              this.$message({
                type: "info",
                message: "已取消!",
              });
            });
        }
    </script>
```

4.3.2 支付订单金额

(1)算法设计思路

用户如果提交订单成功，则会跳转到支付界面，用户需要先支付订单金额。这里调用了 Alipay 方法将前端传过来的金额信息放入到 Alipay 的方法里面，使其跳转支付页面，可使用扫码或支付密码两种方式支付。支付完成后跳转到首页。

(2)实现代码

```
//支付宝沙箱配置
   private final String APP_ID = " 2022000121667943";
   private final String APP_PRIVATE_KEY = "MIIEvAIBADANBgkqhkiG9w0BAQEFAASCBKY
wggSiAgEAAoIBAQCj+c7UYkC7cFocSDzpkvu8rMwHGPqWOutDZOTRGo/3yKrSNY5kfWasy
QctXMP87JHiFsIkfYUFT5H8zwGXE40YeiUdAIpxfg0EN2127WEC4YjdozVIW/4JDM9UE5cU+
E3QUS9//9xjqATkJ27c9vHa7XJtDpPe1RQci4QlUREmTgmJ4/W1AgklkR16zHHYZq27UVrZohw
xjfMNuHOXi4jpwD8m51Z0UYYx1pUgv365mI6mQaBjYy4Y28BlWMeE31DlAnKqwPaaXW1V
EdyL+IYanxAXQEJbWb9td8qZCZkiSdRG1Kj4nljZyF7eT85kwpmHwTUtFnllxR4tX272ZIb/AgM
BAAECggEAGOkGRrV2aAM1bT58Im+11n/ZrupYhyIEQ4S8Thfe7yGI8c45B4MVagJQGNAS5g
V6VAJpE+oNnzTrVKCN+Ul7EdDcen9uiLdBevaApqSBLFLe2+zDV27CNNXBWWBwfzsJdzAe
```

Ky/PZ0RLGe3zpZXOQBf6Xh7sz9lG6N85AAJEvLC35J7D9tfxDDXgFv5jKeSMK4jauUwSyMuH
KaDfekaVWxn7IL45opiERDZwyS8QS3mpDmLEdB62PkJtFBAaKjI7IlNLWFh+VbLzWhVy0xXa
V1ad1UGdJfDTtUMIqeyJR3S5Tml1JQ7DitiMjg82s3jxh+BmBgMYc6QFk2mBcgQKBgQDXyapa
CbHBEdw4xnBYUePJ7N7KR9oEY92LaBifAnw7GoFBiAiuQkAgZD0DsmCvS6ih/ZQcf328gLq+
N5X5m9EV7yoxpi3z2AkEbQ9AsG/wCKLbjM9TjW+zb3KfCzlIG5snAFtwV/0uonI2LbNcdQ/oG
OmbLfnaWYLh5YcQKBgQDCiGkcZjL4q8AYub9MBNJXg3ERFG9ulrTUIG3rIP02HDJip5PScm
X4wrsShU9yfQGZ9tmrNWB/Wf0hR3GYrS9q4BSV4WcgbQIye2NPLQM8BLvWbK76vSJCNW
VMUhGi0KD4EmJQb8Z81aPmMZxUEymgL8wDiiJMWHmRYWIO18YObwKBgHg+KM3pTw
K92BBPC3U7lIoFkxP2u9bsaxxukiGw+dbZayNAEU5BHoiuMo2nzVaA+2Pg8LJ8nGz48pZiVRD1
MapeTg3CgyK4xDWSLAipCIch2NYsfa8aRh8fwCg90fmFPU4ZNlzyoAsmSwKlIdNTVZe/POS2
HS2AGfufVhIZyNERAoGAW6tydMqILHyq6jvltIn1IP40ssrBdAiar/eUNJ/4Ep6Y87pqQ1pbSS1ig
GjHtnjH32jey/o8PecHDL6g/kRmOuPbb4GnWjVkOqvAcZgRlR/Z8EKI9mlYEoEc+ikWXHYZ1K
y9HZB+pWb7dEdelpPtIqpDZkuavAVZTDPqxHnTwAsCgYAmSMQvgE7AfSnrHRltA+eZ1GRrC
dGVzP/YoupCYUZE5e+nrd+l57pr/CqvqjMoX81e9MadJ3EctrRjz86xygra0wLc9u0W3iYs59hgHd
yjSrVhyBR56IqtwaKyxHaOo3wVk7erxmxLbiqNsFdBXkQlvCTPRJxMj33a2hvwlXc+lg==";
private final String IPAY_PUB_KEY = "MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCg
KCAQEA1tCYzJegJNkt8qHft/meEromV27foTd+Pf2HjKBzaDRAYlqmvvuCN2hPQbo0Whni364
UntcjF+qXr6NJAq55fc9+g6JqpfjxC0wqsFSnyoi76HbJLQD4VpkpNp2bcgyQfxxIF4hqXWoDLmB
1K2kJOvXZq4ishyKkfC/KTIDo7qTtqKCNwDB1XvcKOnIjLm7FfSptGttnGqkLNgQmkQqNOXA
6Hk3Aw2uLR41dlcoYaaAxI4Hcz28QblgOZ8Ljj4pyuTgqdRkXaJMSxHbcQwrXalHEvkJVJv7Nqv
nEfE7djAV7uw+K1eyh1wURtblSyEYntW0LaJDZY4NAYUKrxVDt8QIDAQAB";
//沙箱接口路径,正式路径为 https://openpi.alpay.com/gateway.do
private final String GATEWAY_URL = "https://openapi.alipaydev.com/gateway.do";
//支付宝同步通知路径
public static String return_url = "http://localhost:8080/#/frontIndex";

4.4 预购中心模块的设计与实现

4.4.1 添加产品至预购中心

(1)算法设计思路

用户在登录账号后，可在导航栏中浏览理财产品信息，点击查看详情进入产品详情页面，并将产品添加至预购中心中，用户可以查看当前用户的预购意向产品和预购产品数量和金额，并可以通过多选框选择预购产品，然后单击结算按钮选择支付方式等。

(2)实现代码

```
    @ApiOperation("添加产品到预购")
    @GetMapping(value = "/addCart/{id}/{pnum}")
    public boolean addCart(@ApiParam("添加产品的参数") @PathVariable("id") String
id,@PathVariable("pnum") Integer pnum, HttpServletRequest request) {
        if(pnum==null){
            pnum=1;
        }
        String username = (String) request.getSession().getAttribute("username");
        List<Product> list= productBiz.CustomerfindProductById(id);
        String pname=list.get(0).getPname();
        Cart cart=cartBiz.findByPname(pname,username);
        String price=list.get(0).getPrice();
        if(cart!=null){
            int pnum1=cart.getPnum();
            int pnum2=pnum1+pnum;
            double sum1=Double.parseDouble(price)*pnum2;
            Cart cart2=new Cart(pname,pnum2,sum1);
            return cartBiz.updateCart(cart2);
        }else{
            double sum=Double.parseDouble(price)*pnum;
            Cart cart1 = new Cart(username, list.get(0).getPimg(), pname,price,
list.get(0).getAveragerate(),pnum,sum);
            return productBiz.addCart(cart1);
        }
    }
```

4.5 后台管理员模块的设计与实现

4.5.1 用户信息管理

(1)算法设计思路

管理员在登录界面输入账号信息登入后端管理系统，可对用户信息进行管理。单击左侧导航栏的“用户管理”，页面数据在 Mounted 生命周期函数中渲染，使用 Axios 来调用用户控制器里面的获取所有用户信息的方法，将返回的 List 信息以 JSON 格式传输到前端，页面定义一个新的集合，将返回的数据中的集合赋予该新集合，通过遍历获

取每一条用户信息。

(2)实现代码

```
@ApiOperation("用户信息接口")
  @GetMapping(value="/findUser/{index}")
 public Map<String, Object> findUser(@ApiParam("分页的参数") @PathVariable("index")
Integer index){
        if (index==null){
            index=1;
        }
        int size=5;
        List<User> list = userBiz.findUser(index, size);
        int count = userBiz.countUser();
        Map<String, Object> map = new HashMap<String, Object>();
        map.put("list", list);
        map.put("index", index);
        map.put("count", count);
        return map;
    }
```

4.5.2 角色信息管理

(1)算法设计思路

管理员在登录界面输入账号信息登入后端管理系统，可对角色信息进行管理。单击左侧导航栏用户管理中的“角色管理”，页面数据在 Mounted 生命周期函数中渲染，使用 Axios 来调用用户控制器里面的获取所有用户信息的方法，将返回的 List 信息以 JSON 格式传输到前端，页面定义一个新的集合，将返回的数据中的集合赋予该新集合，通过遍历获取每一条角色信息。管理员可点击修改角色信息，会弹出模态框，管理员对原有信息进行修改，然后点击提交，Axios 将管理员提交的修改信息作为参数保存到数据库中，刷新界面，数据回显。

(2)实现代码

```
@ApiOperation("修改角色接口")
    @GetMapping(value="/updateRole/{rid}/{rname}/{rdesc}")
    public boolean updateRole(@ApiParam("修改的参数") @PathVariable("rid") String rid,
                              @PathVariable("rname") String rname,
                              @PathVariable("rdesc") String rdesc,
                              HttpServletRequest request
```

```
                                    ) {
        String rauthor=(String) request.getSession().getAttribute("username");
        Role role=new Role(rid,rname,rdesc,rauthor);
        System.out.println(rid);
        return userRoleBiz.updateRole(role);
    }
    @ApiOperation("添加角色接口")
    @GetMapping(value="/addRole/{rname}/{rdesc}")
    public boolean addUser(@ApiParam("添加角色的参数") @PathVariable("rname") String
rname,
                           @PathVariable("rdesc") String rdesc,
                           HttpServletRequest request
                           ) {
        String rauthor=(String) request.getSession().getAttribute("username");
        Role role=new Role(rname,rdesc,rauthor);
        return    userRoleBiz.addRole(role);
    }
```

4.5.3 理财产品管理

(1)算法设计思路

管理员登录账号进入后端管理系统，可对平台理财产品信息进行管理。单击左侧导航栏的“理财产品管理”，页面数据在 Mounted 生命周期函数中渲染，使用 Axios 来调用理财产品模块控制器里面的获取所有理财产品信息的方法，将返回的 List 信息以 JSON 格式传输到前端，页面定义一个新的集合，将返回的数据中的集合赋予该新集合，通过遍历获取每一条理财产品信息。Axios 将该领养申请信息的 Id 作为参数传递到后端系统

(2)实现代码

```
    @ApiOperation("模糊查询产品")
     @GetMapping(value = "/findProductMo/{index}/{pname}")
     public Map<String, Object> findProductMo(@ApiParam("模糊查询产品的参数")
 @PathVariable("index") Integer index,@PathVariable("pname") String pname) {
        if (index == null) {
            index = 1;
        }
        int size = 3;
```

```
        List<Product> list = productBiz.findProductMo(index, size, pname);
        Map<String, Object> map = new HashMap<String, Object>();
        //调用查询总记录的方法
        int count = productBiz.countMo(pname);
        map.put("list", list);
        map.put("index", index);
        map.put("pname", pname);
        map.put("count", count);
        return map;
    }
```

4.5.4 理财产品收益率管理

(1)算法设计思路

管理员登录账号进入后端管理系统，可对产品收益率进行管理。单击左侧导航栏的“理财产品管理”，页面数据在 Mounted 生命周期函数中渲染，使用 Axios 来调用产品收益率模块控制器里面的获取所有产品收益率信息的方法，将返回的 List 信息以 JSON 格式传输到前端，页面定义一个新的集合，将返回的数据中的集合赋予该新集合，通过遍历获取每一条产品收益率信息。Axios 将该产品的 Id 作为参数传递到后端系统，调用产品收益率模块控制器里的修改方法，根据 SQL 语句从数据库修改该信息。

(2)实现代码

```
@ApiOperation("根据 id 删除近期收益率接口")
    @GetMapping(value="/delete/{rid}")
    public boolean deleteProduct(@ApiParam("删除的参数") @PathVariable("rid") String rid) {
        System.out.println(rid);
        return recentlyRateBiz.delete(rid);
    }
    @ApiOperation("修改近期收益率接口")
    @GetMapping(value="/update/{rid}/{jan}/{mar}/{may}/{july}/{sep}/{nov}")
    public boolean update(@ApiParam("修改的参数") @PathVariable("rid") String
rid,@PathVariable("jan")String jan,@PathVariable("mar") String mar,
      @PathVariable("may") String may,
      @PathVariable("july") String july,
      @PathVariable("sep")String sep,@PathVariable("nov") String nov) {
       RecentlyRate recentlyRate=new RecentlyRate(rid,jan,mar,may,july,sep,nov);
        return recentlyRateBiz.update(recentlyRate);
```

```
}
```

4.5.5 预购中心管理

(1)算法设计思路

管理员登录账号进入后端管理系统，可对平台预购中心进行管理。单击左侧导航栏的“预购中心”，页面数据在 Mounted 生命周期函数中渲染，使用 Axios 来调用预购中心信息模块控制器里面的获取所有预购中心信息的方法，将返回的 List 信息以 JSON 格式传输到前端，页面定义一个新的集合，将返回的数据中的集合赋予该新集合，通过遍历获取每一条预购信息。管理员删除预购记录，将删除预购的 id 作为参数用 Axios 传递到预购信息模块的控制器中并调用删除的方法，修改数据库，回显数据。

(2)实现代码

```
@ApiOperation("批量删除产品接口")
    @ResponseBody
    @GetMapping(value="/delCartAll/{ids}")
    public boolean delCartAll(@ApiParam("删除的参数") @PathVariable("ids") String[] ids) {
        return cartBiz.delCartAll(ids);
    }
    @ApiOperation("根据 id 删除产品接口")
    @ResponseBody
    @GetMapping(value="/deleteCart/{cid}")
    public boolean deleteCart(@ApiParam("删除的参数") @PathVariable("cid") String cid) {
        return cartBiz.deleteCart(cid);
    }
```

4.5.6 订单管理

(1)算法设计思路

最高权限管理员登录账号进入后台管理系统，可对订单信息进行管理。单击左侧导航栏的“订单管理”，页面数据在 Mounted 生命周期函数中渲染，使用 Axios 来调用订单模块控制器里面的获取所有订单信息的方法，将返回的 List 信息以 JSON 格式传输到前端，页面定义一个新的集合，将返回的数据中的集合赋予该新集合，通过遍历获取订单信息，Axios 将管理员提交的修改信息作为参数保存到数据库中，刷新界面，数据回显。

(2)实现代码

```
    @ApiOperation("根据 id 完成订单接口")
@GetMapping(value = "/updateTwo/{oid}/{username}")
```

```
public boolean updateTwo(@ApiParam("完成的参数") @PathVariable("oid") String oid,
                         @PathVariable("username") String username,
                         HttpServletRequest request) {
    String author=(String) request.getSession().getAttribute("username");
    Order order=new Order(oid,author);
    List<HashMap<String, Object>> list=userBiz.findByUsername1(username);
    String mid= (String) list.get(0).get("MONEYID");
    Money money = userBiz.findByMoneyid(mid);
    double Totalmoney=money.getTotalmoney();
    double pay=money.getPay();
    double income=money.getIncome();
    double balance=money.getBalance();
    Order order1=orderBiz.findById(oid);
    double totalsum=order1.getTotalsum();
    double exincome=order1.getExincome();
    orderBiz.updateTwo(order);
    orderBiz.updateSuccess(oid);
   double    recentTotalMoney=Totalmoney+exincome;
   double recentPay=pay+totalsum;
   double recentIncome=income+exincome;
   double recentBalance=balance+exincome;
   Money money1=new Money(mid,recentTotalMoney,recentIncome,recentPay,recentBalance);
   boolean b= userBiz.updateMoney(money1);
    return b;
}
```

4.5.7 订单详情管理

(1)算法设计思路

管理员登录账号进入后端管理系统，可对用户提交的订单信息进行管理。单击左侧导航栏的“订单管理”，页面数据在 Mounted 生命周期函数中渲染，使用 Axios 来调用订单详情模块控制器里面的获取所有订单详情的方法，将返回的 List 信息以 JSON 格式传输到前端，页面定义一个新的集合，将返回的数据中的集合赋予该新集合，通过遍历获取每一条用户提交的订单详情信息。管理员点击该条订单的详情按钮，可以查看用户提交订单的具体购入产品细节。

(2)实现代码

```
@ApiOperation("根据订单 id 查询接口")
    @GetMapping(value = "/findDetailById/{index}/{id}")
    public Map<String, Object> findDetailById(@ApiParam("分页的参数")
@PathVariable("index") Integer index,@PathVariable("id") String id) {
        if (index == null) {
            index = 1;
        }
        int size = 3;
        List<Detail> list = orderDetailBiz.findDetailById(index,size,id);

        int count = orderDetailBiz.countDetailById(id);
        Map<String, Object> map = new HashMap<String, Object>();
        map.put("list", list);
        map.put("index", index);
        map.put("count", count);
        return map;
    }
```

4.6 本章小结

经过对本吾爱理财平台进行系统的设计分析，目前已经进入编码阶段，为了提高网站的质量，在编写程序时应该注意开发和代码的规范化，逻辑应该很清楚，不要有太多的冗余代码，注重性能的提高，注释要完善清晰，以便于后期的维护与开发。本吾爱理财平台主要包括了理财产品分类、首页介绍、登录、个人中心、预购中心、用户、管理员等模块。此章节着重介绍了本吾爱理财平台各个部分功能实现的设计思路与设计细节，以及实现功能的核心代码实现。

5 本吾爱理财平台的运行与效果分析

5.1 网站运行效果

本吾爱理财平台从使用角色上划分，可分为用户角色和管理员角色；用户角色又可划分为前台界面、领养动物、寄养服务、志愿者申报、动物信息、交流这六个部分，管理员角色模块可进行用户、管理员、动物、领养记录、寄养等信息的管理工作。

5.1.1 前台界面模块运行效果

(1)首页展示界面

在网站主页游客或用户可以选择想要查看的内容，导航栏放有“首页”、“股票”、“基金”、“债券”、“银行储蓄”、“保险”、“黄金”、“个人中心”、“登录”九个板块内容的链接，用户单击想要了解的内容即可跳转到该内容页面，见图 5-1。

图 5-1 首页导航栏展示图

(2)查看理财产品板块内容

在理财产品板块，游客或用户可查看本平台的所有理财产品，包括“股票”、“基金”、“债券”、“银行储蓄”、“保险”、“黄金”等六大理财产品版块。在该板块可查看产品的大概信息，具体见图 5-2。

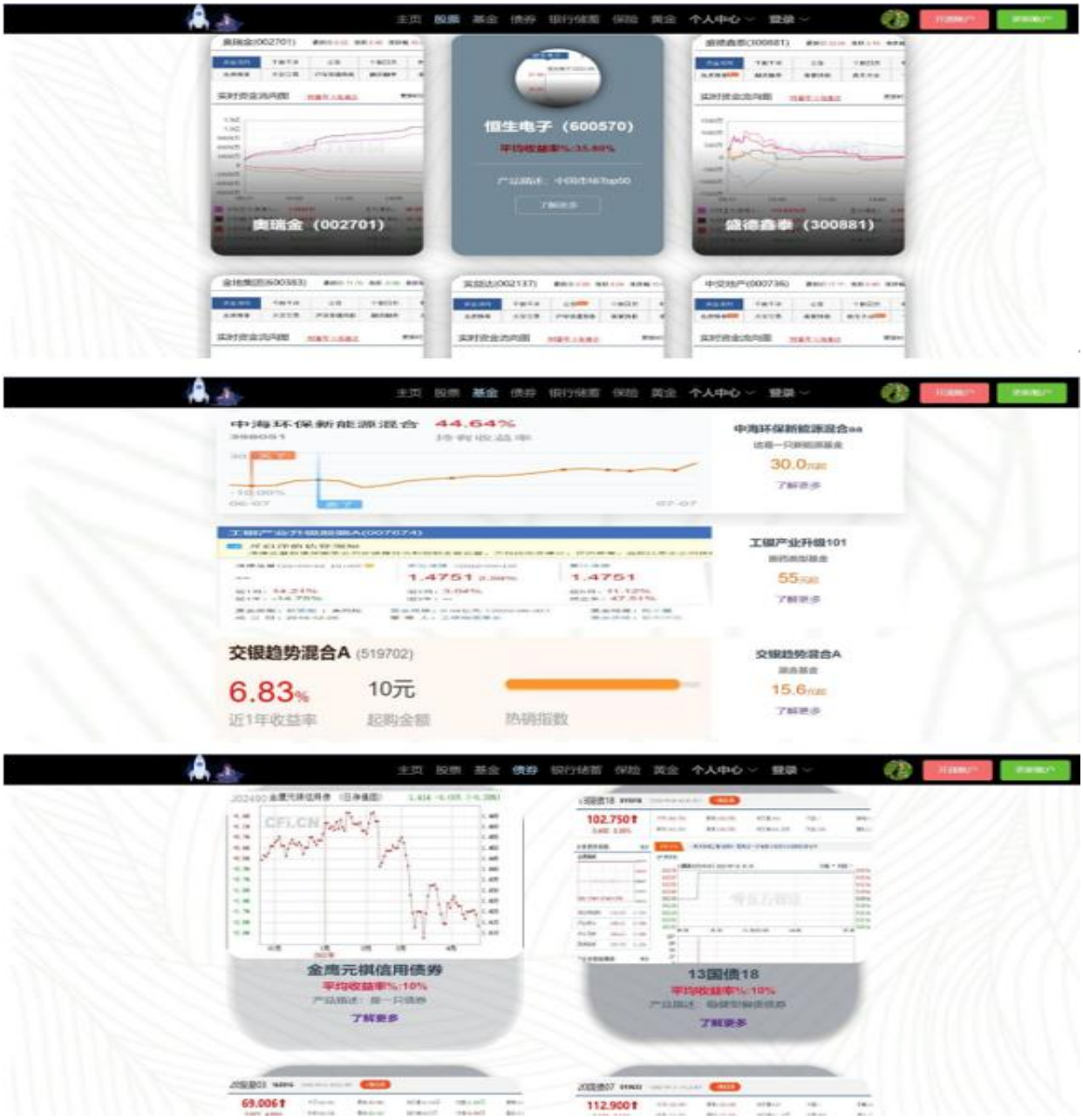

图 5-2 理财产品板块示例图

(3)查看“理财产品详情”板块内容

在“理财产品详情”板块，游客或用户可通过单击进入详情操作。左侧是产品类型介绍视频，右侧展示理财产品细节，可选择加入预购或者直接结算，并弹出窗口输入购入数量。具体见图 5-3

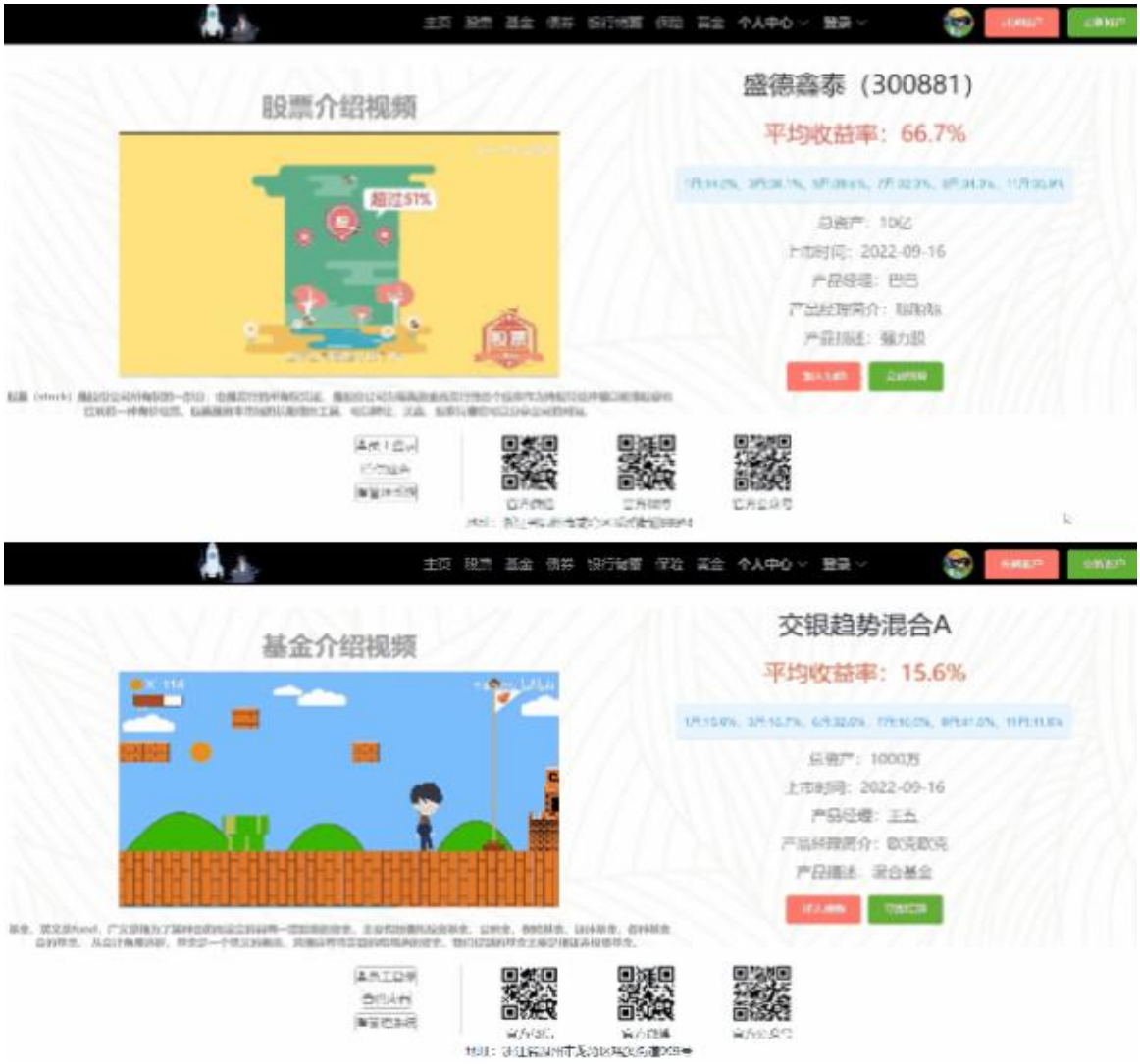

图 5-3 “股票详情、基金详情”板块示例图

5.1.2 用户中心模块运行效果

(1)用户注册

游客需注册账号方能成为本网站的用户，直接注册，用户输入用户名、密码，单击提交即可完成注册。并且弹出提示：注册成功！如果注册的用户名已存在，则会提示相应的错误。完成注册之后，平台会自动跳转到登录页，用户可进行登录。详情见图 5-4。

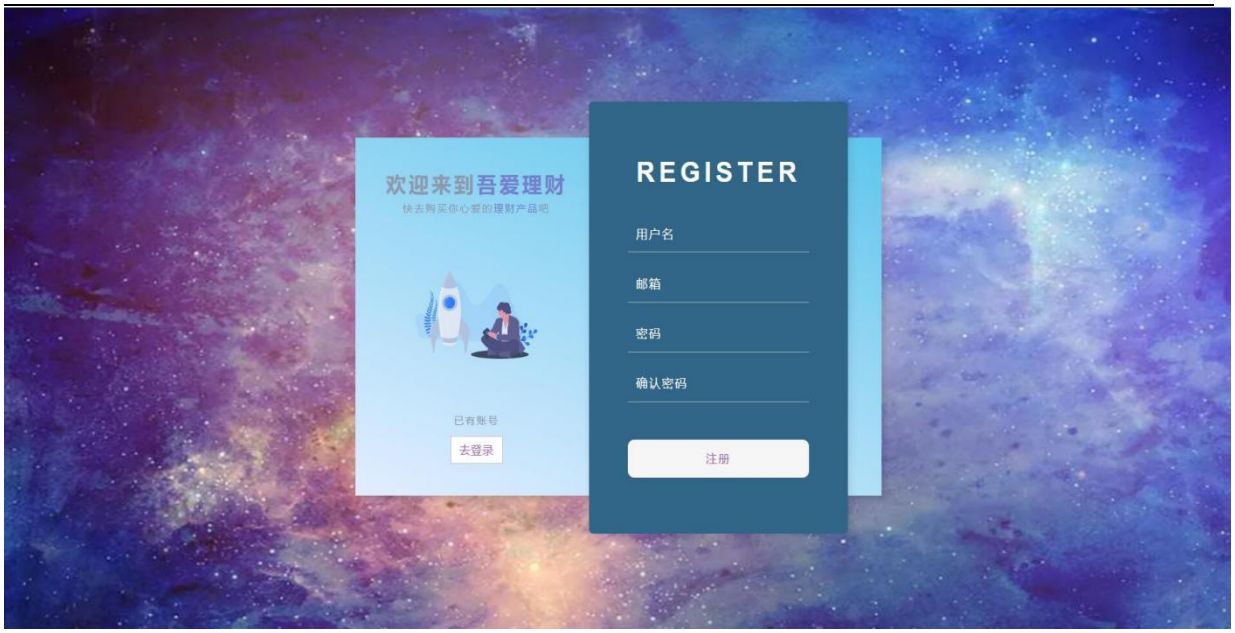

图 5-4 用户注册示例图

(2)用户登录

网站普通用户在成功注册后，可点击去登录跳转到登录界面，用户只需要输入用户名和密码，单击登录，并进行登录的图形验证，如果验证则会提示登录成功。若用户输入信息有误，则无法登录详情见图 5-5。

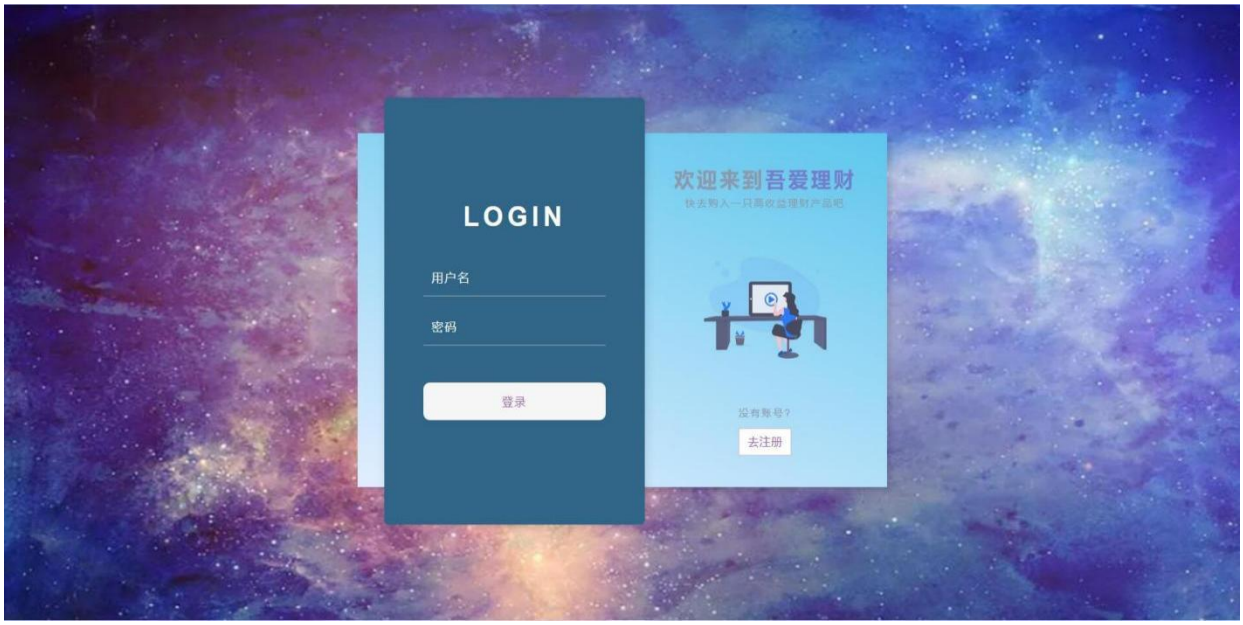

图 5-5 用户登录示例图

(3)查看“个人中心-个人详情”板块内容

在“个人详情”板块，已开通账户的用户可查看余额，如果余额不足，可点击充值按钮进入支付页面充值个人余额，左侧饼状图显示的是该登录用户的余额以及收入支出情况，右侧显示的为该用户的订单状态及详情信息。具体见图 5-6

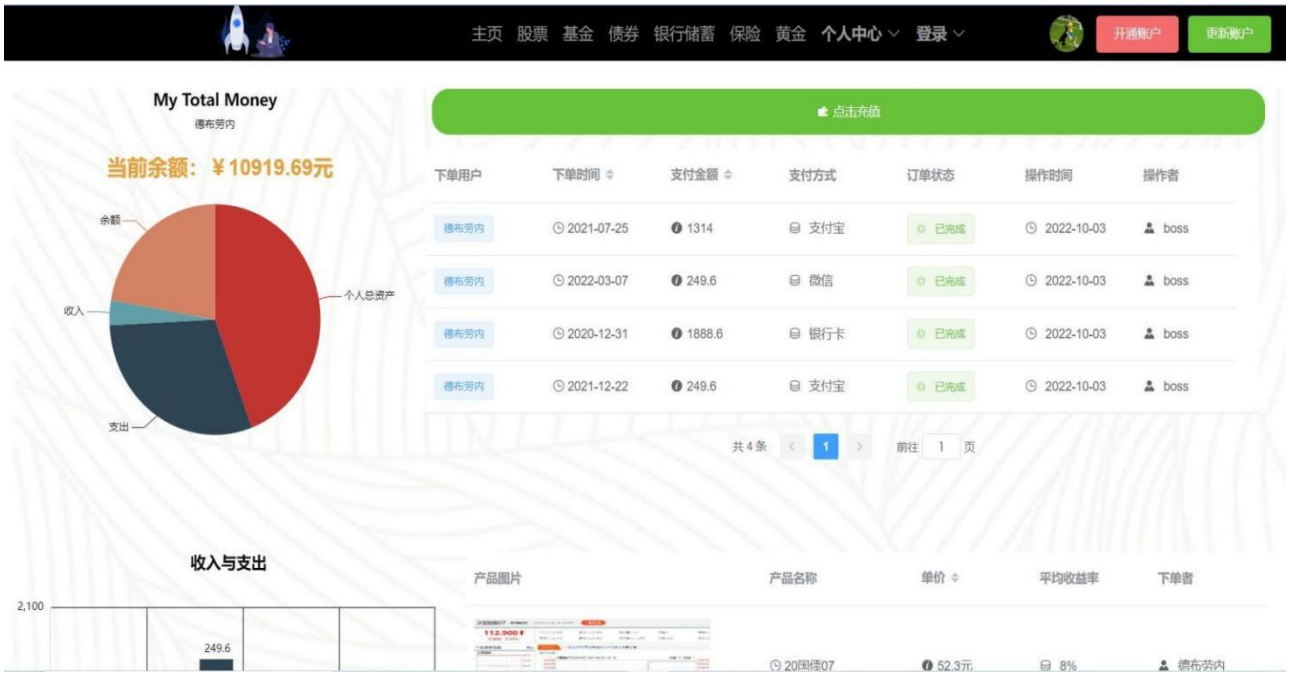

图 5-6 “个人详情 1”板块示例图

(4)在“个人详情”板块的第二部分，其中左侧柱状图展示的是该用户在不同时间段购买的不同订单的收入支出状况，右侧表格显示的是该用户具体购买过哪些产品，并做了去重处理。具体见图 5-7

图 5-7 “个人详情 2”板块示例图

(5)个人中心板块修改个人信息

用户登录完成后可以进入个人中心-信息修改页面，本平台对密码做了隐藏处理，可点击展示，并修改昵称、密码、性别等其他信息，同时也可以点击上传个人头像对图片信息进行修改，见图 5-8。

图 5-8 用户查看并修改个人信息示例图

5.1.3 预购模块运行效果

(1)查看“个人中心-预购中心”板块内容

在“预购中心”板块，已开通账户的用户可查看已选购的产品以及预购的数量并且可以对数量进行修改，同时可以批量的删除产品，清空选择等。在进行以上这些操作的同时，右下角的预购总额都将实时变化，若想结算则点击结算进入支付，若想继续购物可以点击继续购物，继续购入。具体见图 5-9。

图 5-9 “预购中心”板块示例图

5.1.4 订单模块运行效果

(1)直接结算功能

选择一支产品直接结算（不加入预购中心），进入页面后首先可以查看结算的产品信息，查看总价，并选择服务星级，以及支付的方式。如果用户余额不足以去支付订单，则会提示余额不足请充值。具体见图 5-10。

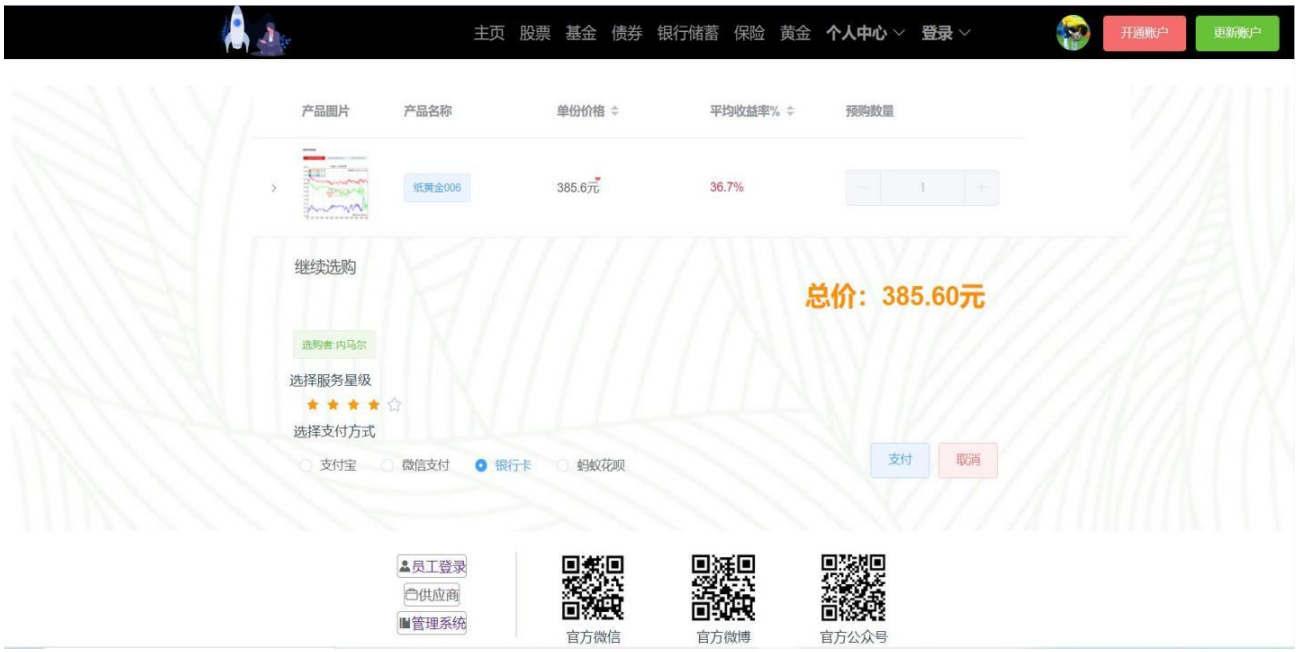

图 5-10 直接结算功能示例图

5.1.5 后台管理模块运行效果

(1)管理员登录

管理员输入其账号的用户名、密码，并进行登录图形验证，此处设有表单验证，也可以点击重置按钮对信息进行重置，同时也可点击左侧 Go 首页进入前台门户登录页。若信息有误，会弹出提示框提示登录失败，见图 5-11。

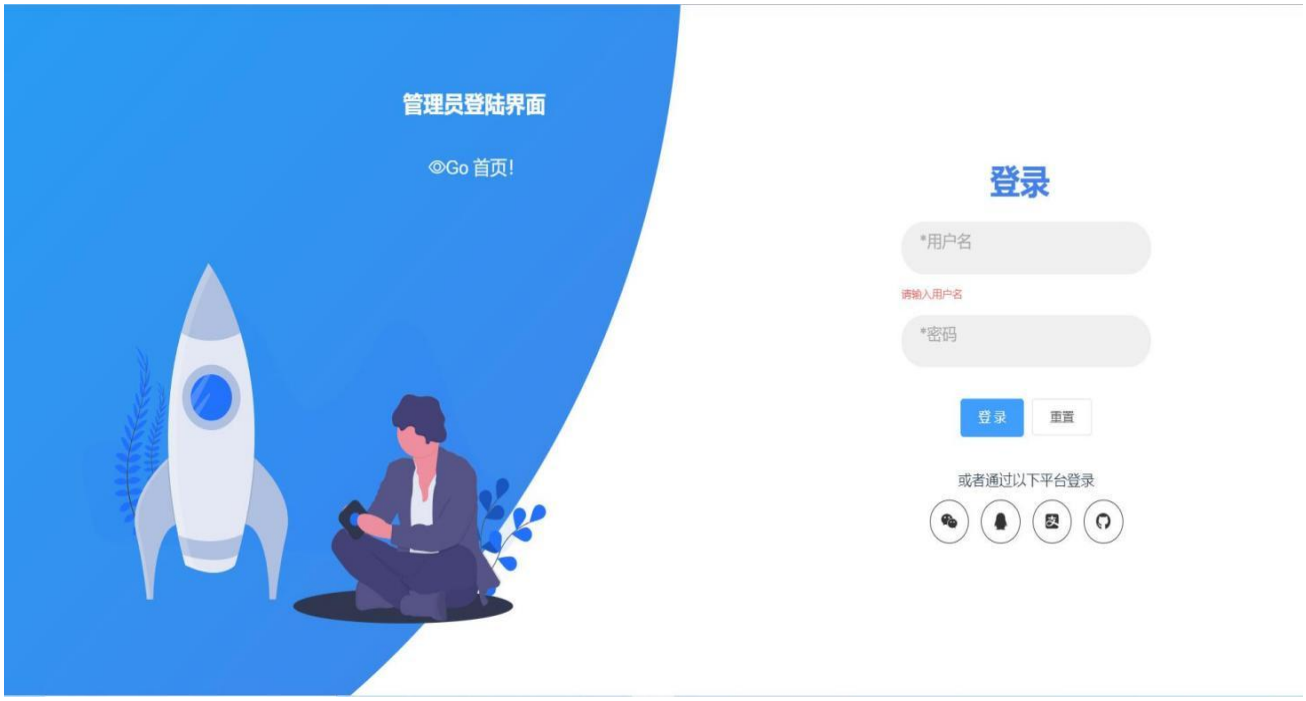

图 5-11 管理员登录示例图

(2)后台管理首页 1

管理员成功登录后自动跳转到后台管理首页，第一栏显示本地实时天气信息，第二栏展示所有已在本平台注册的用户余额信息，第三栏展示随机产品的每月收益率，第四栏显示该系统设计的流程任务。见图 5-12

图 5-12 后台管理首页 1 示例图

(3)后台管理首页 2

管理员成功登录后自动跳转到后台管理首页，在后台首页的第二个板块，通过展示了一个折线图图表显示了随机五只理财产品的不同月份的平均收益率，同时可以点击某一支产品进行隐藏和展示。具体见图 5-13

图 5-13 后台管理首页 2 示例图

(4)用户管理列表

权限最高的管理员登录后可查看所有登记在册的用户信息，权限最高的管理员可以查看、修改、和删除所有下级员工和普通用户的信息，并且还设置了修改状态，当将用户状态禁用后，该用户将无法登陆，并且可以通过详情按钮查看该用户的收入、支出、余额、总资产等关键信息，见图 5-14。

图 5-14 用户列表示例图

(5)用户添加

在用户添加中仅限最高权限的管理员可进行添加管理员/普通用户的功能，并且根据提示只需要设置一些必要信息，其余信息已经自动生成，如果需要修改可进入个人中心或者用户列表中修改个人信息。见图 5-15

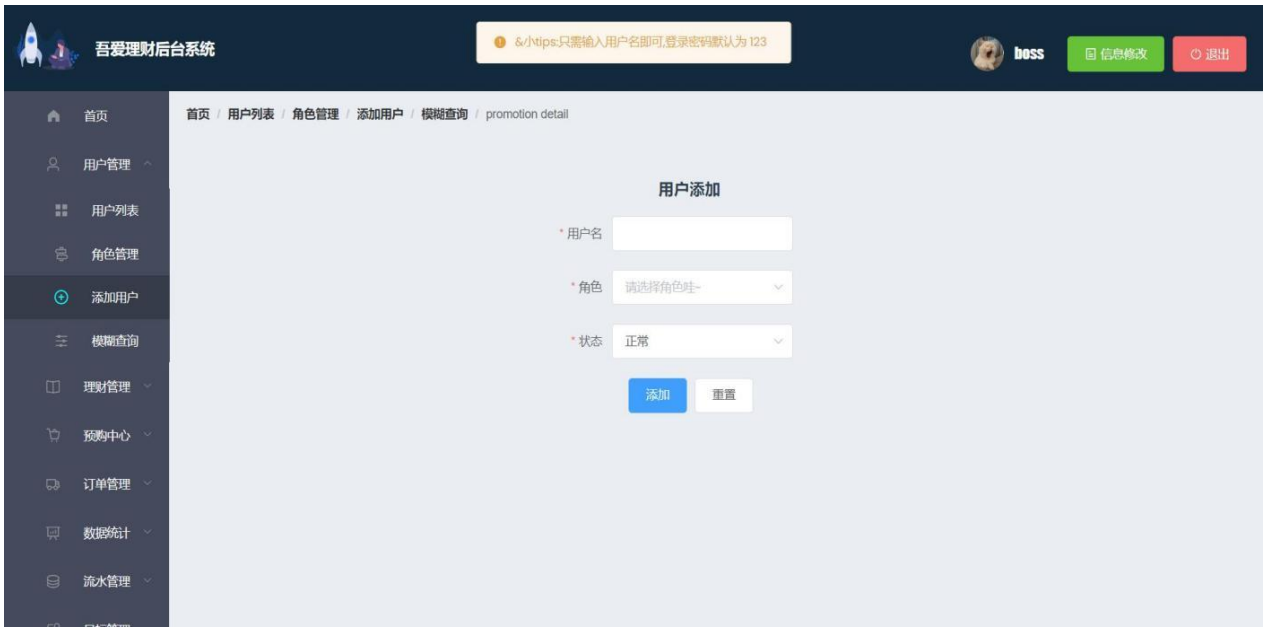

图 5-15 用户添加示例图

(6)角色权限管理列表

在用户管理的角色管理中权限最高的管理员可以添加角色，对角色的信息进行修改和删除。在修改后该条记录会显示是被哪一位管理员所修改的。以及该条信息被修改的当前时间。具体见图 5-16。

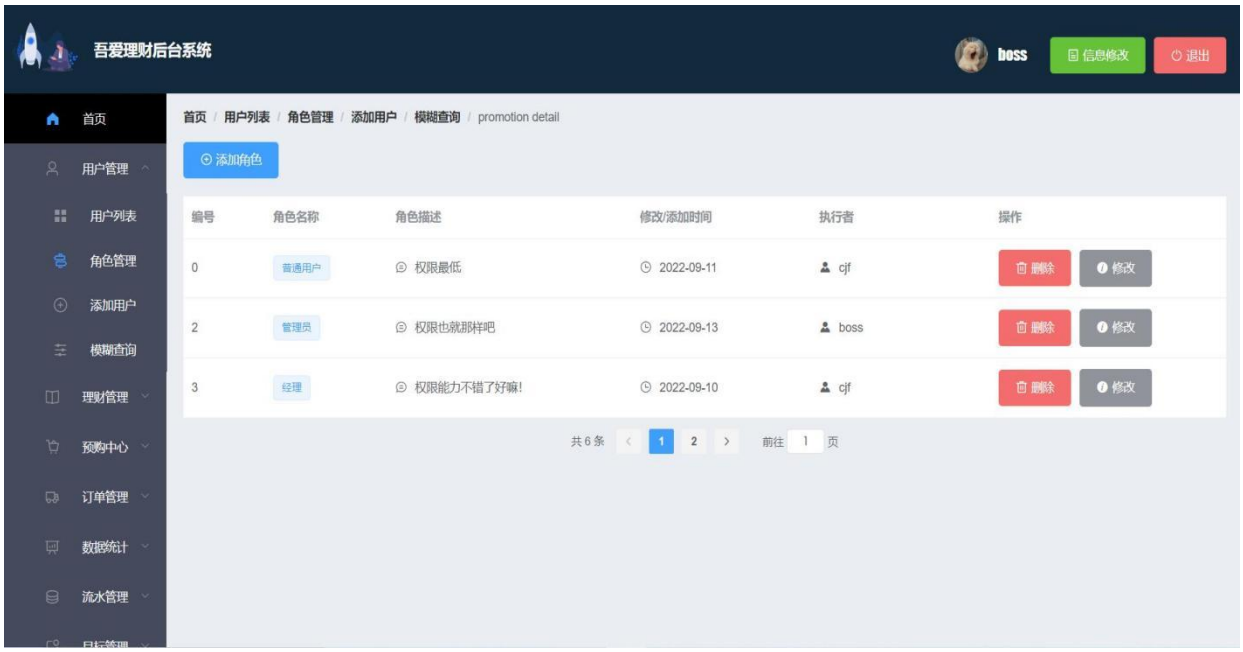

图 5-16 角色权限管理示例图

(7)理财管理列表

设计类似于用户管理列表，左侧箭头可展开内容查看理财产品的细节描述，同样可对产品状态进行控制，当产品状态被禁用时，前台门户将不会展示该产品；并且低权限的管理员只能进行查看，不允许对产品进行编辑内容和删除记录。同时可对产品内容进行模糊的查询，方便对某一支产品的查看。见图 5-17。

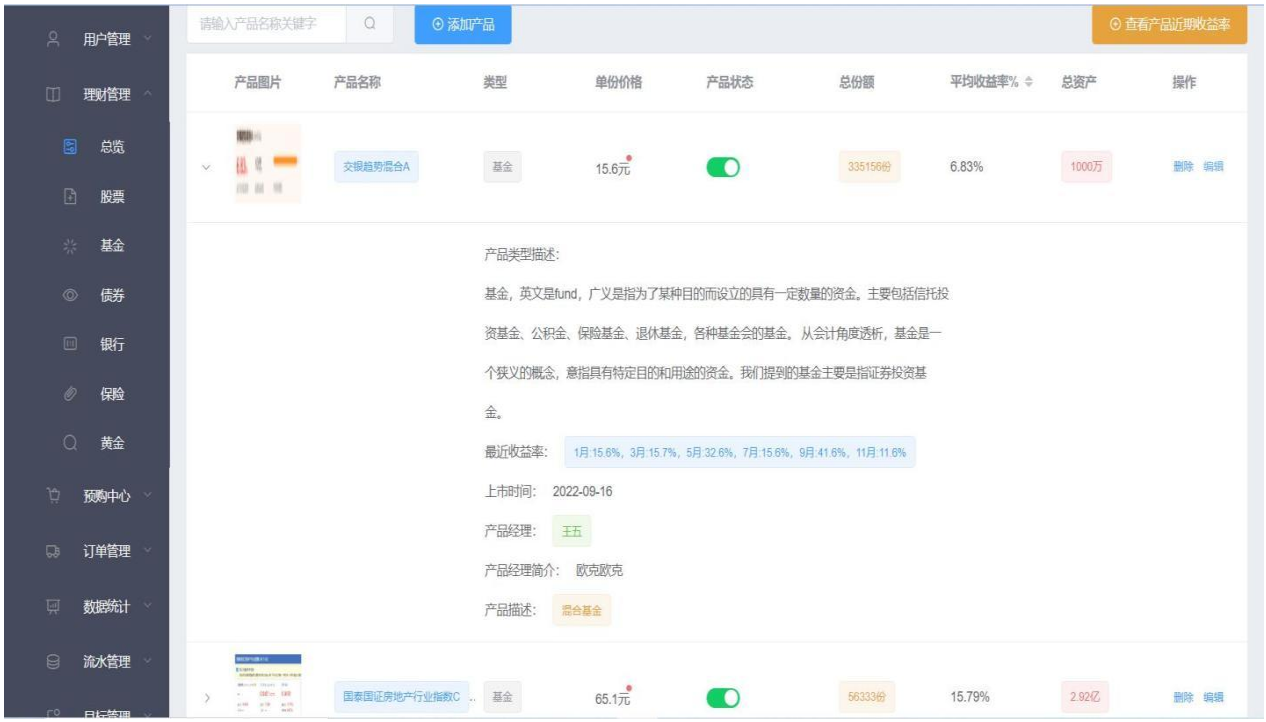

图 5-17 理财管理示例图

(8)理财产品收益率管理

点击查看产品近期收益率按钮后可展示所有产品的近期收益率，当添加一支新的产品后，会默认生成一个对应的产品收益率编号，并且最高权限的管理员可对某一支产品的近期收益率进行编辑数据。图 5-18

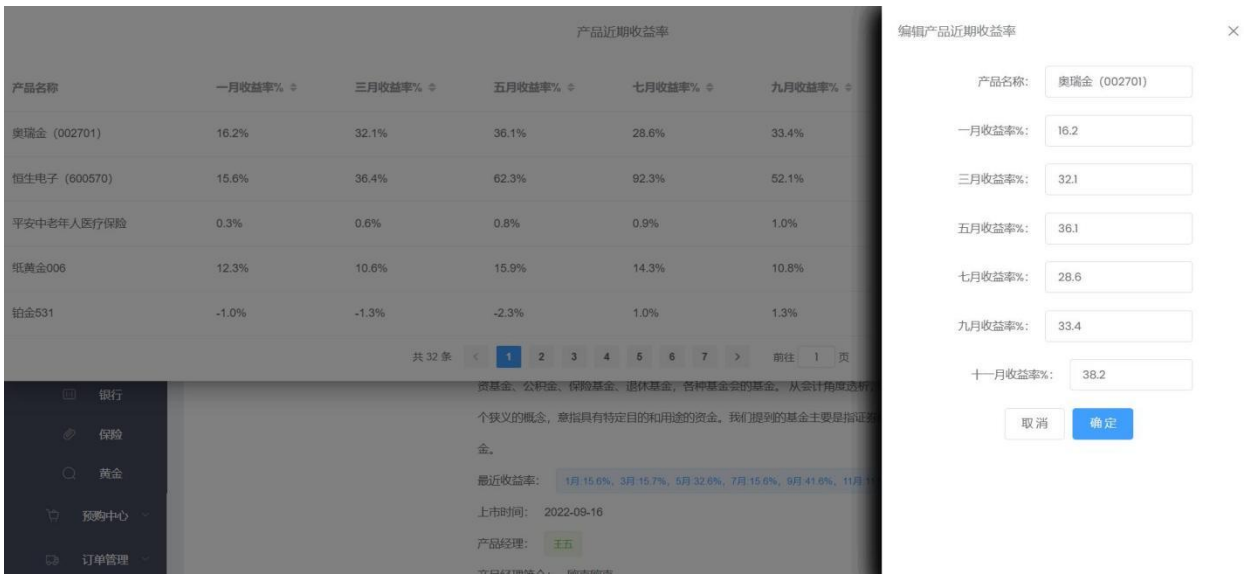

图 5-18 理财产品收益率管理图

(9)理财分类管理

根据产品类型查看不同类型的产品。在页面中首先可以看到左上角的产品类型，其次可查看该类型产品的图片，名称，类型，平均收益率，状态，以及产品总资产等等信息，具体见图 5-19。

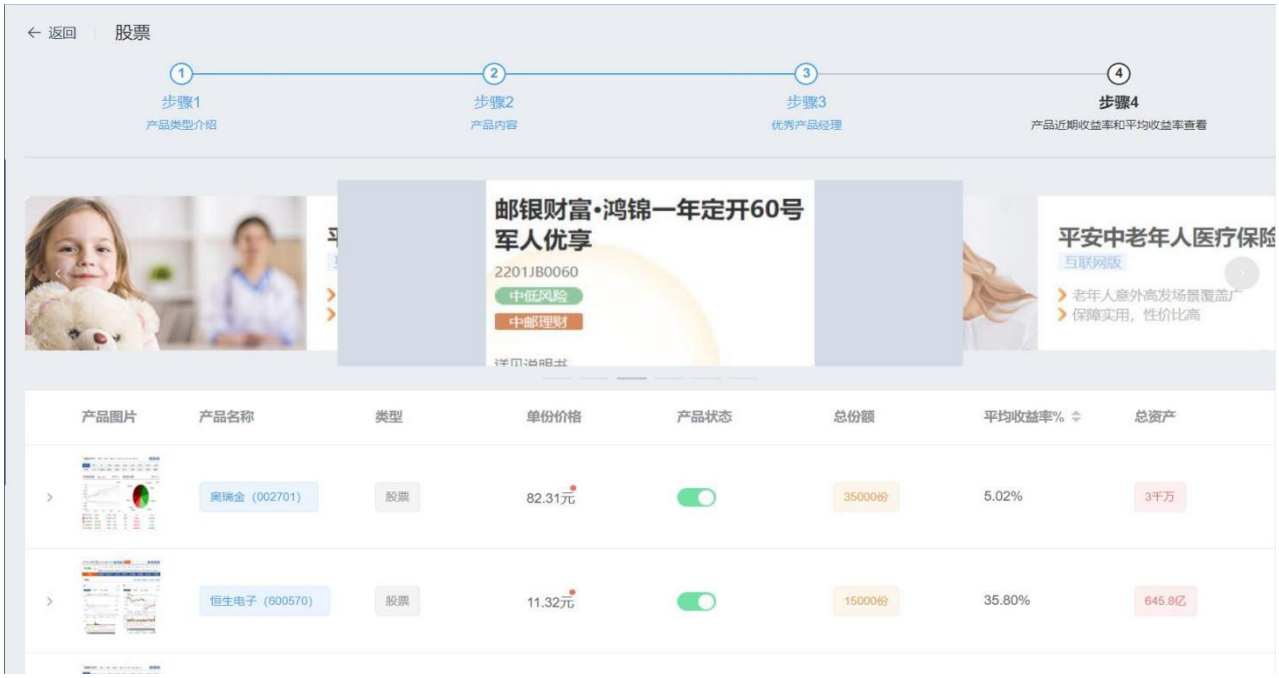

图 5-19 理财分类管理图

(10)预购中心管理列表

管理员可根据预购者查询其预购的内容，并可对预购单条记录进行删除，同时也实现了批量选择删除的功能。并实现了分页的功能。如果想要根据查询该用户下的预购信息，可以通过左上角搜索框输入用户名查看。具体见图 5-20。

图 5-20 预购中心管理示例图

(11)订单管理列表

权限最高的管理员可对订单的详情进行查看、对订单记录进行删除；并实现了批量删除的功能。当普通用户在前台页面支付完成一条订单后发送给后台让管理员进行处理，拥有完成订单功能权限的管理员可点击完成订单为每一位用户结算该订单的金额，当该订单完成后，完成按钮将变成灰色，保证同一条订单不被重复完成。具体金额会根据当前产品平均收益率计算，结算完成后，普通用户能在个人中心查看个人资产和余额的变化，管理员则可点击用户金额查看。见图 5-21。

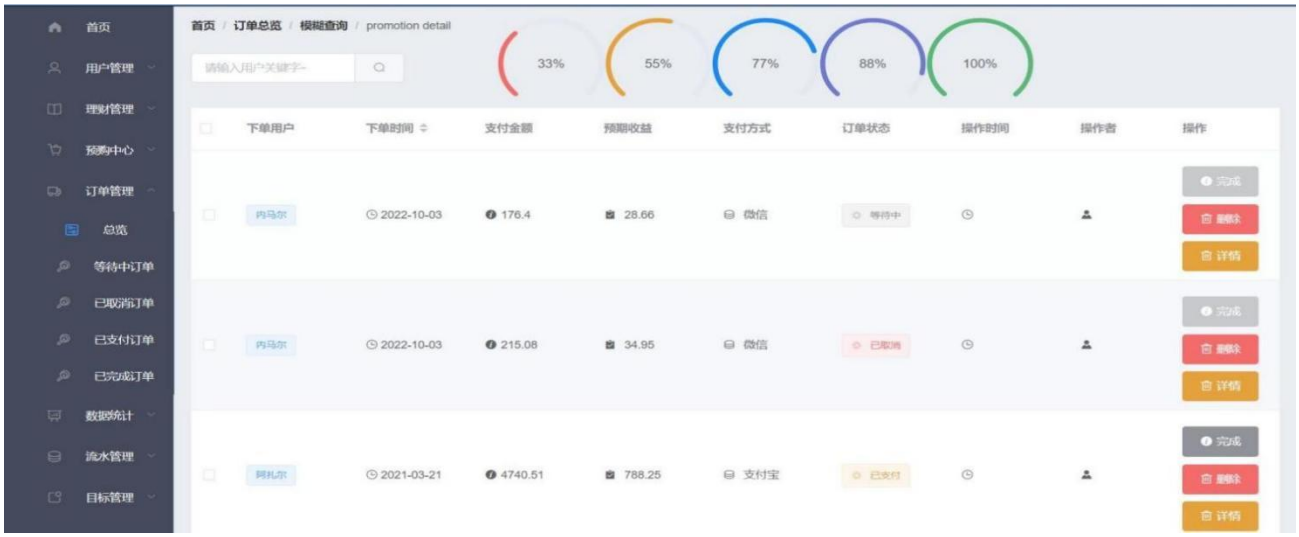

图 5-21 订单管理示例图 1

(12)订单管理详情列表

管理员可点击订单详情查看该条订单购入产品的具体内容细节。并且实现了分页查看的功能，以及显示订单购买的产品，产品的名称，产品的单价，产品的收益率和下单者。具体见图 5-22

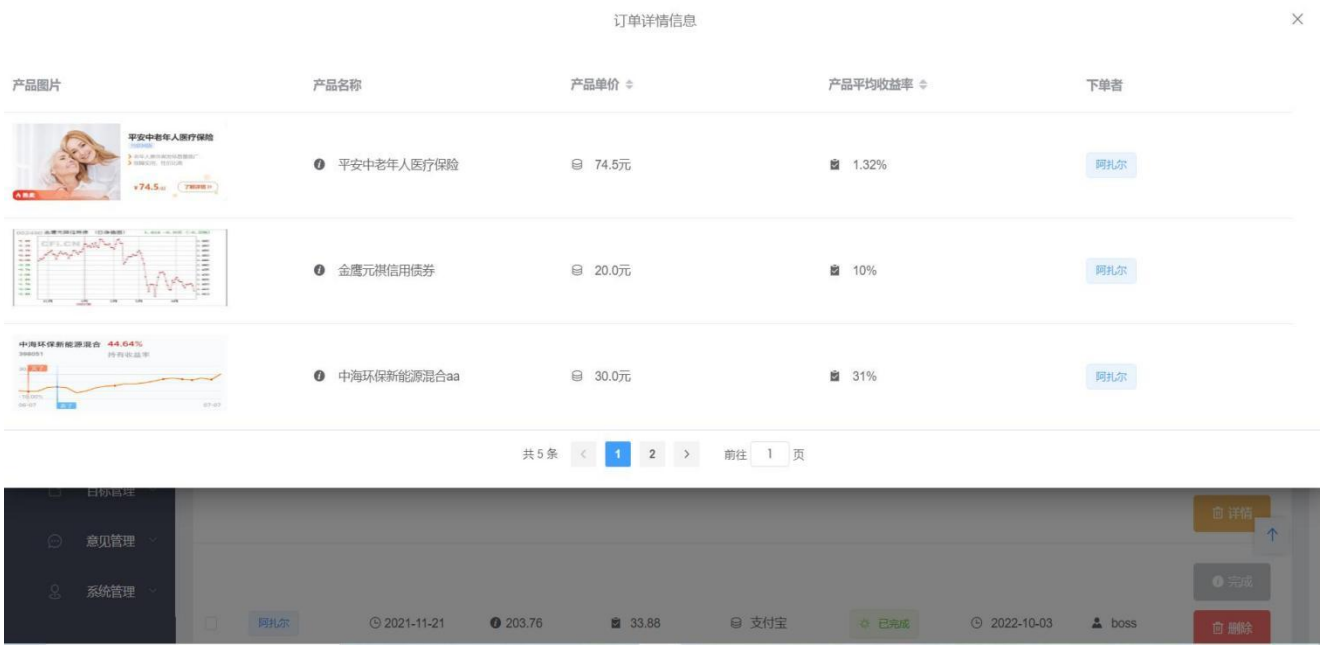

图 5-22 订单详情示例图

(13)订单分类管理列表

下图显示的是已完成状态的订单，管理员点击完成后修改，自动将操作时间填入、操作此条订单的当前管理员填入，在该状态下由于订单状态已是已完成状态，因此不允许对订单进行重复完成操作，同时也能对已完成的订单进行批量删除的功能。见图 5-23

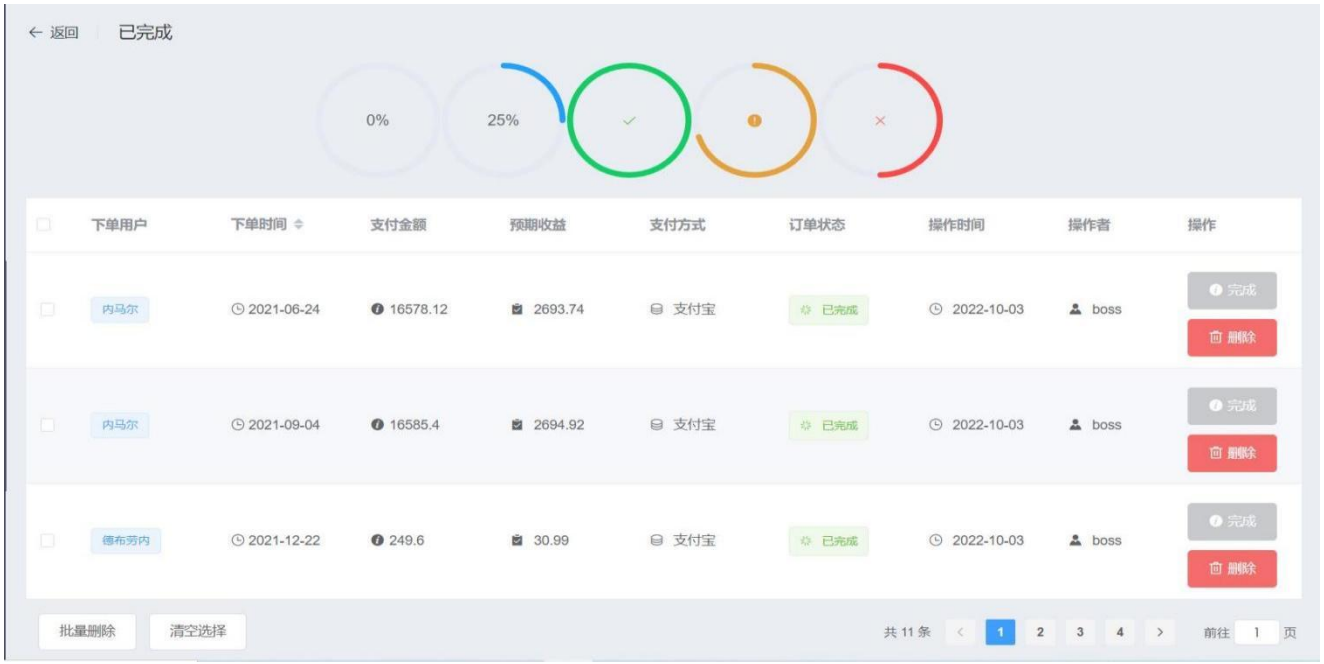

图 5-23 订单分类管理示例图

(14)数据统计列表

管理员可在此查看本平台所有的数据图表，供页面显示。其中下图的图表显示的是该平台所有注册用户的资产情况，包括收入支出以及余额信息，并通过折线图和柱状图的形式更好的进行数据的展示。具体见图 5-24。

图 5-24 数据统计示例图

5.1.6 支付模块运行效果

(1)用户支付订单金额或者充值个人余额界面展示

本网站目前只支持支付宝支付，用户进入支付页面，用户可直接扫描左侧付款码支付费用，也可通过填写支付宝账户信息，登录支付宝账号，并完成支付操作，支付完成则会跳转回首页，继续购物，详情见图 5-25。

图 5-25 用户支付订单或者充值余额示例图

5.2 本章小结

本章节对于本吾爱理财平台的各个模块的功能及内容做了文字描述和示例图展示，较为详细地讲解了在本网站的大概流程。总结来说，对于前端用户可进行的操作有：用户注册、登录；查看前端界面模块内容；根据导航栏选择相应的理财产品进行查看；点击产品详情进入产品详情页；在详情页可以选择将产品加入预购或者直接结算，并在点击的同时输入购入数量；结算的方式有两种：一种是通过预购中心进行批量选择产品进行结算，另一种直接点击一支产品进行结算生成订单；最后生成订单去支付；对于后端管理系统的管理员来说，可进行的操作有：对用户信息，用户角色信息，理财产品信息，用户预购中心信息，数据统计信息等类别进行查看、批量删除和批量添加操作；其中只有最高权限管理员才可以对订单的功能进行删除和完成，普通管理员只能查看订单。管理员可通过后台管理首页的柱状图、折线图、饼图查看动态数据，也可对用户的资产进行分析和操作。本章节的内容相当于是给用户和管理员提供了一份十分详细、简便明了的使用说明书。

6 软件测试

6.1 测试简介

6.1.1 产品简介

本吾爱理财平台是为用户提供了一个更快捷、方便的理财平台，前端用户可通过本网站上挑选心仪的理财产品，线上提交订单，用户可通过本网站了解很多理财知识，尽可能的给用户带来更好的使用感。后端管理系统有着操作简单、模块划分明确的特点，方便管理员对用户信息、理财产品信息、订单信息等的管理，提高工作效率。

6.1.2 测试目的

对本吾爱理财平台进行测试是为了检测网站是否能够正常运行，是否适应当前大多数电脑系统或浏览器，功能设计是否具有人性化，能否满足用户需求等，其中包括：

(1)网站能否正常无错运行，功能是否完全实现；

(2)业务流程是否正确；

(3)数据传输过程是否完整、数据传输是否具有安全性和保密性等；

(4)能否满足网站使用者需求，操作是否简单方便。

6.1.3 测试范围

(1)这一部分对本吾爱理财平台的模块测试范围进行了概述，模块名称包括用户、用户信息、管理员、余额充值、加入预购、形成订单等，该测试模块的优先级都为 1 级，测试的范围详情见表 6-1。

表 6-1 测试范围表(网站模块)

应用角色名称	模块名称	描述	优先级
吾爱理财平台	用户	用户能够顺利登录注册，用户成功登录后进入个人中心。	1
	用户信息	编辑用户名、昵称、邮箱、电话等；	1
	管理员	管理员可对用户、理财产品、订单、预购记录等信息进行管理。可对个人信息及密码进行修改操作	1

续表 6-1

应用角色名称	模块名称	描述	优先级
吾爱理财平台	余额充值	用户发现个人余额不足以支付订单时，可以通过在用户个人中心界面充值余额。	1
	加入预购	用户可在产品详情中了解该产品，并将填写预购数量将该产品加入预购中心。	1
	形成订单	用户可多选理财产品，或者直接购入单支理财产品确认总价，形成订单并支付	1

(2)这一部分对本吾爱理财平台的阶段测试的范围进行了概述，主要的测试分为集成、界面、易用性、兼容性、以及性能和安全性的测试，主要面向的是网页的基本单元模块的测试，具体测试的范围详情见表 6-2。

表 6-2 测试范围表(测试阶段)

测试阶段		描述	优先级	重要级
集成测试		一些单元模块虽能独立工作，但是当几个单元连接起来，就不能保证可以正确运行，集成测试具有全局观，可使用集成测试来检查连接起来的模块是否能够正确运行，数据可否完整传输。	3	2
系统测试	界面测试	本网站是要面向所有人，因此，要检测网站是否适应当下主流浏览器。界面测试主要是检查在各浏览器界面上能够顺利运行，且网站图片、菜单等符合要求等。	3	2
	易用性测试	要保证本网站使用者能够快速上手，需要易用性测试来检查网站的使用难度，是否是简单易上手的。	4	2
	兼容性测试	本网站是要面对所用人，要求在各类电脑、浏览器上都能运行。兼容性测试就是要检测在各电脑上是否兼容。	3	2
	性能测试	进行性能测试来检测网站的各项性能指标是否达到标准。结合负载测试和压力测试进行。	3	2
	安全性测试	安全性测试即要测试网站运行时，数据的传输过程是否存在安全隐患。	3	1

6.2 测试进度

在本章节部分，会将测试进度以表的形式呈现，测试进度分为计划开始、实际开始、和实际结束的三个方向进行进度测试。对需求、功能设计、模块集成、以及系统等方面进行测试，并在最后填写测试报告。具体信息见表 6-3。

表 6-3 测试进度表

一期			
测试过程	计划开始日期	实际开始日期	实际结束日期
了解测试需求	2023/4/4	2023/4/4	2023/4/5
制定网站测试计划	2023/4/5	2023/4/5	2023/4/5
设计功能测试用例	2023/4/6	2023/4/6	2023/4/8
模块测试	2023/4/7	2023/4/7	2023/4/9
集成测试	2023/4/7	2023/4/7	2023/4/9
系统测试	2023/4/7	2023/4/7	2023/4/10
回归测试	2023/4/8	2023/4/8	2023/4/11
网站测试结果报告	2023/4/9	2023/4/9	2023/4/12

6.3 测试资源

6.3.1 人力资源

这一章节对测试工作进行了人力划分，主要分为测试工程师、性能测试人员、安全性测试人员，根据网站的功能需求制定测试的分析，通过对测试结果的分析，从而找到问题所在。具体信息见表 6-4。

表 6-4 人力资源表

角色	测试人员	具体职责或注释
测试工程师	陈锦房	根据本网站的功能需求制定测试计划和测试方案；搭建测试环境；编写测试用例；记录测试过程；对在测试过程中已出现的问题或潜在问题要记录并分析讨论，找到解决方案。
性能测试人员	陈锦房	要求对网站的性能进行分析测试，从网站的负载能力、轻度以及压力问题这三个方面入手，编写测试用例，记录测试过程，对于出现的问题要能够及时找到解决的方法并进行跟踪检测。
安全性测试人员	陈锦房	安全性测试包括程序、数据库安全性测试，要求测试人员从这两面入手，确定测试计划，编写安全性测试用例，对测试结果进行分析，找到问题所在，跟踪监测，保证网站安全性。

6.3.2 测试环境

此章节对测试的环境进行了简单说明，软件编译器主要采用IDEA 2021.2.3，数据库采用的是Oracle, 硬件环境采用的是PC机，安装内存 16GB，64 位操作系统。可用硬盘空间 250GB, 详细信息见表 6-5。

表 6-5 测试环境表

软件环境：IDEA 2021.2.3
测试环境：IDEA 2021.2.3
测试管理工具环境：Windows10, Tomcat8.0,JDK1.8
数据库环境：Oracle
硬件环境：PC 机，安装内存 16GB，64 位操作系统
处理器：AMD Ryzen 7 3750H with Radeon Vega Mobile Gfx 2.30 GHz
测试工具环境：3.28Ghz Cpu，内存 8G, 可用硬盘空间 89G
数据库环境：3.64Ghz Cpu，内存 8G, 可用硬盘空间 110G

6.4 测试策略

6.4.1 数据和数据库完整性测试

本章节对数据和数据库的完整性进行测试，测试目标主要分为测试范围、测试技术、开始标准、完成标准、以及测试重点和优先级。对数据库的访问保证安全性和保密性，确保数据库信息不受损。详情见表 6-6。

表 6-6 完整性测试表

测试目标	数据完整传输，保障数据的安全性和保密性
测试范围	使用者注册登录模块、个人中心模块、预购中心模块、形成订单、余额充值等。
测试技术	对数据库的访问的方法和进程进行调用，填入正确或错误数据，检查数据库，看数据库能否正常运行，检查数据返回是否正确。
开始标准	提交测试的版本，从实际执行测试的时候开始。
完成标准	数据库访问和进程均正常，数据库未受损。
测试重点和优先级	重点关注注册登录模块、个人信息模块、后台管理模块、预购中心模块、余额充值等。

6.4.2 集成测试

集成测试是单元测试的扩展，简单来说，集成测试就是把两个通过检测的模块组合成一个，测试它们之间的接口，监测它们的组合能否成功正常运行，本网站的集成测试

具体信息见表 6-7。

表 6-7 集成测试表

测试目标	网站业务能够正常进行
测试范围	确保网站能够完整正确运行，若有流程未实现，需及时记录并寻找方法处理。重点测试网站核心模块。
测试技术	根据数据的准确性对不同模块的进行测试，并测试各模块的功能是否完善。使用白盒测试和黑盒测试交叉检测，将数据反复进行处理，针对系统集成进行全面且透彻的检查。
测试重点和优先级	本吾爱理财平台的核心功能点作为测试重点，具有高优先级，先进行检测，且重点检测。
需考虑的特殊事项	进行相应操作时，网站可提供友好界面提示。

6.4.3 用户界面测试

本章节从测试目标、测试范围、测试技术、开始标准、完成标准、需考虑的特殊事项进行对用户界面进行测试，重点测试每个窗口是否可以正确浏览，页面是否符合使用者的需求，具体信息见表 6-8。

表 6-8 界面测试表

测试目标	界面模板是否正常；图片、菜单栏能否正确展示；页面跳转是否正常；窗口之间、字段之间和快捷键的使用等，这些都要符合要求
测试范围	用户端所有页面
测试技术	测试窗口，是否每个窗口均可正确浏览
开始标准	二期的网站测试开始介入时
完成标准	页面符合使用者需求
需考虑的特殊事项	一些界面操作需设置使用者权限

6.4.4 兼容性测试

本吾爱理财平台是要面向所有人的，因此进行兼容性测试来测试本网站在不同的操作系统平台上、不同浏览器上能否正常运行是必要的，在测试技术方面同样要求做到兼容，具体测试内容见表 6-9。

表 6-9 兼容性测试表

测试目标	在不同系统、浏览器中所有的功能能否正常使用。
测试范围	网站需兼容 Edge、IE、谷歌、Firefox 等浏览器；可在不同电脑系统下正常运行。
测试技术	兼容性测试
开始标准	二期网站测试开始介入时
完成标准	在不同系统和浏览器中均可正常运行
测试重点和优先级	核心功能点能够正常运行
需考虑的特殊事项	无

6.4.5 性能测试

性能测试在软件质量保证中起到重要作用。性能测试的内容有很多，这里针对网站的负载能力、是否运行多人同时使用、页面的响应时间、轻度以及压力问题进行测试，详情见表 6-10。

表 6-10 性能测试表

测试目标	确保网站性能良好
测试范围	测试范围包括用户、理财产品、后台管理等模块。
测试技术	多人登录前端用户界面；网页反应时间
开始标准	二期网站测试开始介入时
完成标准	网站可允许多人同时在线进行加入个人预购中心、形成订单、个人余额充值等操作，且不发生错误。
测试重点和优先级	是否运行多人同时使用；页面响应时间。

6.4.6 安全性和访问控制测试

进行安全性测试是为保护用户账号和重要数据不被泄露而采取的。访问控制测试是要测试控制运行的有效性，根据用户类型设置访问和功能权限，保障网站能够正确的执行流程。测试详情见表 6-11。

表 6-11 访问控制表

测试目标	低级安全性：部分页面根据使用者类别进行权限设置。
测试范围	所有网站使用者：用户和管理员
测试技术	根据用户类型设置访问和功能权限。
开始标准	二期网站测试开始介入时
完成标准	不同使用者有不同的访问和功能使用权限
测试重点和优先级	用户注册登录信息格式判断

6.5 测试风险

进行测试工作，风险是不可避免的。本章节对在测试工作中遇到的风险做了描述，主要分为时间资源、人力资源和其他三个方面进行描述。主要先保证功能能够正常运行，再进行测试工作的进行，详情见表 6-12。

表 6-12 测试风险表

序号	测试风险	风险描述	解决办法	影响程度
1	时间资源	本吾爱理财平台测试的周期较短	先保证核心功能正确运行，部分测试工作二期进行。	高
2	人力资源	测试人员无专业测试经验	参考有价值的测试文档进行学习	高
3	其他	其他未知风险	代码异常处理机制	未知

6.6 测试用例

测试用例的基本要素包括测试用例编号、测试标题、重量级别、测试输入预期结果、测试结果等，主要分为用户注册登录、预购中心、订单、以及后台管理的标题描述。详情见表 6-13。

表 6-13 测试用例表

<table>
<tr><td colspan="6">测试用例</td></tr>
<tr><td colspan="2">项目名称</td><td colspan="2">吾爱理财平台</td><td>程序版本</td><td>V1.0</td></tr>
<tr><td colspan="2">模块名称</td><td colspan="4">用户注册登录、个人中心、预购中心、后台管理</td></tr>
<tr><td colspan="2">设计人员</td><td colspan="2">陈锦房</td><td>编制时间</td><td>2022/10/20</td></tr>
<tr><td colspan="2">功能特性</td><td colspan="4">网站不同模块的功能点是否实现</td></tr>
<tr><td colspan="2">测试目的</td><td colspan="4">测试网站不同模块的功能点是否可实现</td></tr>
<tr><td colspan="2">参考信息</td><td colspan="2"></td><td>特殊规程说明</td><td></td></tr>
<tr><td>用例编号</td><td>标题</td><td colspan="2">用例说明</td><td>预期结果</td><td>测试结果</td></tr>
</table>

续表 6-13

测试用例				
项目名称		吾爱理财平台	程序版本	V1.0
模块名称		用户注册登录、个人中心、预购中心、后台管理		
设计人员		陈锦房	编制时间	2022/10/20
功能特性		网站不同模块的功能点是否实现		
测试目的		测试网站不同模块的功能点是否可实现		
参考信息			特殊规程说明	
用例编号	标题	用例说明	预期结果	测试结果
1	用户注册登录	用户在个人信息板块修改用户密码，可进行密码修改，密码统一设为123456。	提示修改成功。	通过
2		用户在个人中心点击余额充值按钮，进入支付金额页面。	页面会跳转到支付界面。	通过
3		用户在登录页面输入注册好的用户名与密码，进行登录	提示登录成功。	通过
4		用户在个人中心的查看个人充值的余额金额	金额充值成功。	通过
5		用户查看个人中心的订单已完成状况。	订单显示成功。	通过
6		用户查看个人中心的数据图表显示	图表数据显示成功。	通过
7		用户将理财产品加入预购中心	提示添加成功	通过
8	预购中心	用户对预购中心产品进行批量删除操作	批量删除成功。。	通过
9		用户多选产品，对产品进行批量结算，进入支付页面	支付成功后，页面会跳转到门户首页。	通过

续表 6-13

测试用例				
项目名称		吾爱理财平台	程序版本	V1.0
模块名称		用户注册登录、个人中心、预购中心、后台管理		
设计人员		陈锦房	编制时间	2022/10/20
功能特性		网站不同模块的功能点是否实现		
测试目的		测试网站不同模块的功能点是否可实现		
参考信息			特殊规程说明	
用例编号	标题	用例说明	预期结果	测试结果
10	形成订单	用户在预购中心通过批量选择产品进行结算形成订单，当个人余额大于订单总额时，方可支付	用户成功跳转到支付页面。	通过
11		用户在产品详情中直接点击直接结算按钮，选择服务星级，确定单条订单信息，进行结算	用户成功跳转到支付页面。	通过
12	管理员管理	管理员登录账号后，可选择用户管理模块，对用户信息进行修改、删除。	提示修改成功、删除成功。	通过
13		最高权限管理员登录账号后，可进行对用户/管理员的添加	提示添加用户成功	通过
14		最高权限管理员登录账号后，可对管理员角色进行管理，包括对角色的删除以及添加	提示删除成功、修改成功。	通过
15		管理员登录账号后，可选择理财产品管理模块，对理财产品信息进行添加、修改、删除。	提示添加成功、修改成功、删除成功。	通过
16		管理员登录账号后，可在左侧导航栏中选择预购管理。	显示用户预购信息。	通过
17		最高权限管理员登录账号后，选择理财产品模块的产品收益率，点击修改按钮对产品收益率进行修改。	提示修改成功。	通过

续表 6-13

测试用例				
项目名称		吾爱理财平台	程序版本	V1.0
模块名称		用户注册登录、个人中心、预购中心、后台管理		
设计人员		陈锦房	编制时间	2022/10/20
功能特性		网站不同模块的功能点是否实现		
测试目的		测试网站不同模块的功能点是否可实现		
参考信息			特殊规程说明	
用例编号	标题	用例说明	预期结果	测试结果
18	管理员管理	管理员登录账号后，可在左侧导航栏中选择订单模块。	显示用户所提交的订单信息。	通过
19		管理员登录账号，可选择不同状态的订单信息	信息显示成功	通过
20		管理员登录账号后，选择订单模块，查看每一条订单的详情，该条订单的具体购入产品细节	订单详细信息显示成功。	通过
21	网站兼容性测试	测试本网站能否在 Window XP 操作系统中的 IE、火狐、Edge 等浏览器中正常显示界面并稳定运行。	页面显示正常、运行无误。	通过
22		测试本网站能否在 Window 7 操作系统中的 IE、火狐、Edge 等浏览器中正常显示界面并稳定运行。	页面显示正常、运行无误。	通过
23		测试本网站能否在 Window 10 操作系统中的 IE、火狐、Edge 等浏览器中正常显示界面并稳定运行。	页面显示正常、运行无误。	通过
24		测试本网站能否在苹果操作系统中的 IE、火狐、Edge 等浏览器中正常显示界面并稳定运行。	页面显示正常、运行无误。	通过
25	易用性	本网站对用户的每步操作是否有友好界面提示。	网站有大量界面提示。	通过
26		本网站窗体大小是否合适。	大小合适	通过

6.7 缺陷报告

本章节是对在测试过程中遇到的问题进行记录并分析缺陷所在，主要分为两个模块个人中心以及个人信息，用户登录账号后可以选择查看个人余额，或者修改个人信息。详细信息见表 6-14。

表 6-14 缺陷报告表

缺陷报告						
软件名称	吾爱理财平台	版本号	V1.0			
测试人员	陈锦房	测试时间	2022/10/20			
浏览器	Edge	操作系统	Windows10			
编号	模块	简单描述	详细描述	优先级	严重度	相关附件
1	个人中心	用户登录账号跳转到个人中心，用户可在个人信息中查看个人余额及订单。	用户登入系统并跳转到个人中心界面；用户可以在个人信息板块查看到其所提交的订单信息，以及用户余额状况	低	低	
2	个人信息	用户登录账号跳转到个人中心，用户可在个人信息中修改密码和其他个人信息	用户登入系统并跳转到个人中心界面；用户可以在个人信息板块查看到自己的信息；用户修改密码，密码可以根据自己的想法决定。	低	低	

6.8 本章小结

本章节通过对本吾爱理财平台的进行测试分析来找到网站的缺陷，提高网站的质量。测试分析工作包括了根据本网站需求制定测试计划并编写测试、集成测试、界面测试、性能测试、安全测试、回归测试等。通过对本网站进行测试分析工作，总结缺陷并跟踪处理问题，且网站能够正常运行，所发现的问题都已经解决，隐藏风险也还在监测中。

7 总结与展望

7.1 总结

本课题阐述了 Java、Spring Boot、Element UI、Vue.js、Oracle[5]在吾爱理财平台中的应用。

首先，Java 语言作为静态面向对象编程语言的代表，程序员以优雅的思维进行复杂编程，使用 Java 开发语言容易上手。在本次的项目中，使用 Spring Boot+Element UI+Vue 部署 Web 服务，Spring Boot 是当前比较火爆的，它使配置变简单，且其内嵌 Servlet 容器，对环境要求不高；Element UI 是基于 Vue.js 的前端框架，它提供了丰富的组件库使得开发者更加方便、更有效率的使用；Vue.js[11]的使用是为了更好的渲染数据。

本吾爱理财平台已经初步实现了项目初期的功能需求，对前端界面的用户来说，已经实现了用户的注册登录，查看前端界面模块内容；根据导航栏选择相应的理财产品进行查看；点击产品详情进入产品详情页；在详情页可以选择将产品加入预购或者直接结算，并在点击的同时输入购入数量；结算的方式有两种：一种是通过预购中心进行批量选择产品进行结算，另一种直接点击一支产品进行结算生成订单；最后生成订单去支付；后端管理系统的管理员也已经实现了对各种数据的管理操作。本网站设置有很多的友好提示，帮助用户和管理员进行操作，以防止操作失误。此外，本网站为保障使用者的账号安全，也做了密码加密的功能，以及管理员等级权限的设置。本网站的操作十分简单，只需简单引导就可快速上手。

本文的研究结论有以下几点：

(1)在做本项目时，使用了一些多样的 Oracle 语句实现想要的功能，虽然学艺不精，但是也顺利地---将本网站较好的完成。

(2)在做前端功能实现时，组件并不适配，通过不断剖析每个属性的含义，逐渐找到了解决的办法。

(3)在做本项目之前，肯定是要对理财方面做全面的调查了解，包括了解国内外理财市场、理财人口基数、市场的调研等等，通过对这些内容做了较为深入的了解之后，我对本网站有了更多的需求分析，对网站功能的功能设计做了很多的完善。

7.2 展望

本吾爱理财平台经过长时间的可行性分析、需求分析、功能设计以及具体功能的实现，目前网站已搭建完成。由于本人网络安全技术方面的缺乏以及网站运维方面了解不多，很多地方还存在着大大小小的问题，本吾爱理财平台还有很多需要完善的地方。

(1)由于缺乏网络安全技术，面对跨站脚本攻击，不能够完全保障使用者的账号安全，并且网站的承载量较低，在线人数过多时将会导致网站崩溃。

(2)本吾爱理财平台使用简单算法，运行效率低下，后期还需要对部分模块的算法进行优化以提高网站运行效率。

(3)本平台还未部署上线，本地项目 Css 图片资源还未压缩，占用空间较大

(4)前端请求 Axios 还未进行封装，请求结构，方法过于单一，文件夹结构较为混乱，还未进行分模块化处理。

(5)本平台目前只进行了简单的测试，对于后期是否会出现问题还是未知的。网站在正式投入使用之前，还需要更多的人工测试和自动测试，也需要有更多用户体验本网站，提出更多宝贵意见。

参考文献

[1] 高岳.互联网金融平台通用清算系统的设计与实现[D].南京大学,2022.DOI:10.27235/d.cnki.gnjiu.2022.001930.

[2] 前端奋斗人.Vue.js 是一个构建数据库驱动的 MVVM 库[J/OL].CSDN.2023.

[3] 何俊峰,朱凌晨.基于 SpringBoot+Vue 实现智慧化工地之动火证审批系统[J].电脑编程技巧与维护,2023(06):127-129.DOI:10.16184/j.cnki.comprg.2023.06.016.

[4] 田松涛,段元梅.基于 SpringBoot 的线上商城平台设计[J].无线互联科技,2023,19(01):56-57.

[5] 张晓达. 民生银行理财产品管理销售系统的设计与实现[D].北京工业大学,2018.董礼.基于 ORACLE 数据库的优化设计研究[J].黑龙江科学,2022,12(10):94-95.

[6] 王萍芳.基于 VUE 的智能采摘机器人前端开发框架研究[J].农机化研究,2023,45(05):229-232.DOI:10.13427/j.cnki.njyi.2023.05.016.

[7] 赵海国.Ajax 技术支持下的 ECharts 动态数据实时刷新技术的实现[J].电子技术,2018,47(03):25-27+57.

[8] 郑玉娟,张亚东.基于 Vue.js 的微商城前端设计与实现[J].信息技术与信息化,2022(11):101-103.

[9] 陈陆扬. Vue.js 前端开发快速入门与专业应用[M].人民邮电出版社:, 201702.207.

[10] 丁勇,周守印,庞英华,黄健.基于 Spring 架构的有机茶园数字化系统构建研究[J].中国茶叶加工,2022(03):21-27.DOI:10.15905/j.cnki.33-1157/ts.2022.03.014.

[11] Teledyne FLIR Unveils Radiometric Vue TZ20-R Thermal Zoom Drone Payload[J]. M2 Presswire,2022.

[12] Snow Software Gets Oracle Database and Oracle Database Options Verification[J]. Wireless News,2023.

[13] LEE Sa Im,YANG Jae Heon,CHAE Byeong Suk,CHO Chong Hyeon,SHIN Tae Yong,LEE Jae Hyeok,PARK Hee Wook,PARK Jeong Suk,KIM Dae Keun. Phytochemical Constituents from the Twigs of Vuex rotundifolia[J],2008.

[14] Fuyuan Cheng. Talent Recruitment Management System for Small and Micro Enterprises Based on Springboot Framework[J]. Advances in Educational Technology and Psychology,2022,5(2).

[15] Gu Mengdie,Sun Rui,Yang Shulin,Gu Huijie,Yuan Ming. Research on copyright appointment registration microplatform system based on vue[J]. MATEC Web of Conferences,2023,355.

[16] Jérôme Proulx. Point de vue sur une perspective canadienne en didactique des mathématiques : réaffirmer le commun et tracer des distinctions[J]. Canadian Journal of Science, Mathematics

致谢

时光如逝，岁月如梭。转眼间在人工智能与软件学院的学习生活就要结束了。今天，一切就要落幕了。对于我而言，在这的求学经历，给予我的不仅是知识上的增长，更重要的是开阔了我的视野，坚强了我的意志，使我的思想变得更加成熟。明天，我将满载着的学识，投身于社会中，为软件开发行业贡献自己的绵薄之力。

首先，要感谢我的父母，和我的姐姐，是他们一直站在我的背后默默的支持我，让我知道，不管我什么时候回头，他们都会在。是父母给了我一个舒适又健康的环境让我茁壮成长，是姐姐在我学习的阶段给予我许多经验和处理困难的方法。在成长之路上，是他们以身作则，教会我成长成人的道理，他们，未曾缺席我人生的每个阶段。我向我的家人表示感谢，也将用实际行动来回报他们。

感谢我的母校——江苏师范大学科文学院，感谢学校的栽培，是它给我们提供优秀的师资力量、良好的教学环境、舒适卫生的宿舍。今后，不管走到何方，都不会忘记母校的教诲，脚踏实地，勤能补拙，为社会贡献自己的力量。

此外，我还要感谢许多学长和同学在整个过程中的帮助和配合。本文是在张茜和曹天杰老师精心指导和大力支持下完成的。张茜和曹天杰老师以其严谨求实的治学态度、高度的敬业精神、兢兢业业、孜孜以求的工作作风和大胆创新的进取精神对我产生重要影响。他们渊博的知识、开阔的视野和敏锐的思维给了我深深的启迪。

最后，特别感谢东软睿道教育，在不平凡的大四这一年，是东软的老师们给予我们在基地生活的温暖，定期举办有意义的活动，生活上为我们提供爱心帮助，课堂上积极为同学们耐心解答，在最关键的就业的时期，联系优质企业，为每一位同学提供最优质的就业方向。这里，我还要感谢东软睿道的周婕婕老师、王伟老师。他们对待每一位学生的细心态度和课堂质量的保证，让我们在大四的这一年在学习之外也收获到了诸多的满足感。