**在线画稿交易平台的设计与实现**

**学生姓名：**

**专 业：计算机科学与技术**

**指导教师：**

**单 位：**

**二○二五年四月**

**摘要**

本文设计并实现了一款在线画稿交易平台，旨在为设计师和普通用户提供一个高效、安全的画稿交易和互动平台。平台支持用户注册、登录、设计师认证、画稿发布、购买交易、余额充值与提现等核心功能。在用户认证方面，平台使用JWT（JSON Web Token）实现安全认证机制；设计师认证采用管理员审核机制，确保平台上的设计师身份真实可靠。平台内的支付功能接入支付宝，用户充值、提现方便快捷。所有的交易过程都能保证资金的安全和顺畅。通过系统测试，平台的各项功能得到了有效验证，系统稳定、功能完备，具备了商业化运作的基础。

**关键词：**画稿交易、JWT、支付、Django

**ABSTRACT**

This paper designs and implements an online illustration trading platform, aiming to provide a high-efficiency and secure platform for illustration trading and interaction for both designers and ordinary users. The platform supports core functions such as user registration, login, designer certification, illustration posting, purchase and transaction, balance recharge and withdrawal. In terms of user authentication, the platform uses JWT (JSON Web Token) to implement a secure authentication mechanism; designer certification adopts an administrator review mechanism to ensure the authenticity and reliability of the designers on the platform. The payment function within the platform is integrated with Alipay, making user recharges and withdrawals convenient and fast. All transaction processes can ensure the safety and smoothness of funds. Through system testing, the platform's various functions have been effectively verified, and the system is stable and fully functional, laying the foundation for commercial operation.

**Key words:** Designer certification mechanism；Draft trading system；JWT；Payment platform；Django；

**目录**

[摘要 I](#_Toc6951)

[ABSTRACT II](#_Toc10992)

[第1章 引言 1](#_Toc17892)

[1.1 研究背景和意义 1](#_Toc3277)

[1.2 国内外研究现状 1](#_Toc19632)

[1.3 研究内容 2](#_Toc31585)

[1.4 研究方法 2](#_Toc24500)

[第2章 开发技术介绍 3](#_Toc28469)

[2.1 Django 3](#_Toc8757)

[2.2 MySQL 3](#_Toc24534)

[2.3 JSON Web Token（JWT） 3](#_Toc29058)

[2.4 Vue.js 4](#_Toc12678)

[2.5 Ant Design Vue 4](#_Toc21586)

[第3章 系统分析 5](#_Toc15765)

[3.1 市场分析 5](#_Toc24139)

[3.2 功能需求分析 5](#_Toc12890)

[3.3 可行性分析 6](#_Toc28806)

[第4章 系统设计 8](#_Toc12328)

[4.1系统架构设计 8](#_Toc18240)

[4.2分区模块设计 10](#_Toc29361)

[4.3交互模式设计 11](#_Toc1570)

[4.4数据库设计 15](#_Toc4787)

[第5章 系统实现 18](#_Toc16327)

[5.1 开发环境搭建 18](#_Toc2534)

[5.2 注册/登录（JWT） 18](#_Toc6757)

[5.3 余额充值 19](#_Toc13188)

[5.4 设计师认证/审核机制 22](#_Toc707)

[5.5 画稿交易 24](#_Toc5602)

[第6章 系统测试 28](#_Toc10213)

[6.1测试的意义及目标 28](#_Toc26092)

[6.2测试方法 28](#_Toc3349)

[6.3功能性测试 29](#_Toc31631)

[6.4测试结论 31](#_Toc11568)

[结语 32](#_Toc26189)

[致谢 33](#_Toc16739)

[参考文献 34](#_Toc2959)

# 引言

1. 研究背景和意义

在当今互联网技术与数字经济高速发展的背景下，传统的艺术作品市场正在经历一场深刻的变革。面对实体市场的边界限制和成本压力，艺术作品交易逐步向线上转移，其中，在线画稿交易平台便是交易形态变革的创新成果。这种转型不仅拓宽了艺术作品的市场范围，还为创作者和消费者提供了更为便捷的交流和购买渠道。在线交易平台的建设已成为解决数字经济时代艺术作品版权保护与合规交易的关键。这不仅涉及知识产权的保护，也对平台的技术架构、安全性、用户隐私保护等提出了严格要求[1][2]。同时，随着社会对于艺术创作的关注提升，年轻创作者逐步进入市场，他们的新颖观点和创作风格也对平台功能设计提出了新的挑战和机遇。

当前，市场上涌现出众多在线画稿交易平台，它们通过提供便捷的交易媒介，促进了版权资源的高效配置。然而，现有平台在用户体验、交易安全性和纠纷解决机制等方面依然存在不少问题。例如，用户在使用平台时，对作品类型、质量的建议权和对管理员的监督权往往未能得到充分保障，对创作者的水平界定也较为模糊，这导致了交易过程中的不透明和不公正现象。

针对这些问题，设计与实现一个高效的在线画稿交易平台显得尤为重要。该平台不仅需要整合多种类型的作品，还应具备良好的用户互动机制，以便用户能够对作品提出建议并参与到纠纷解决过程中。此外，提升管理员的专业素养和服务态度是改善用户体验的关键。

1. 国内外研究现状

随着互联网和电子商务的飞速发展，在线设计作品交易平台逐渐成为设计师与消费者之间的重要桥梁。国内外已经涌现出多种在线画稿交易平台，这些平台不仅提供了设计师展示、销售作品的机会，还为消费者提供了便捷的购买体验。然而，尽管这些平台在一定程度上推动了设计行业的数字化转型，但在实际应用中，依然存在着一些亟待解决的问题和不足。

1.2.1 国际现状

在国际上，许多电子商务平台和在线设计市场已经实现了类似的功能。诸如 Etsy、Dribbble、Behance 等平台，不仅为设计师提供了一个展示和交易的平台，还为消费者提供了便捷的购买体验。这些平台通常采用会员制，通过注册用户、发布设计作品、设定价格以及交易的方式进行管理。虽然这些平台在全球范围内拥有广泛的用户基础和较高的知名度，但依然面临以下挑战：

（1）设计师认证与管理复杂：大多数平台要求设计师提交认证材料，但审核流程往往繁琐且缺乏高效性。平台的审核机制过于依赖人工，容易造成认证时间过长，用户体验不佳。

（2）版权保护不充分：虽然平台提供了一些版权保护措施，但由于设计作品的特殊性，版权问题依然难以避免，盗版与抄袭现象较为普遍，且现有的维权手段较为复杂，缺乏有效的技术保障。

1.2.2 国内现状

在国内，随着互联网技术的不断发展，设计师创作和交易平台逐渐兴起。例如， 站酷、猪八戒网、腾讯云设计市场 等平台，致力于为设计师和创作者提供展示、交易和交流的空间。相比于国外平台，国内的设计市场更多地聚焦于服务本地用户，支持中文界面，并与本土支付系统（如支付宝、微信支付）深度集成，以适应国内用户的需求。国内平台虽然在支付方式、用户界面等方面进行了本地化优化，但仍存在一些不足：

（1）设计师认证机制不完善：尽管国内平台有一定的设计师审核机制，但大多数平台的认证过程仍然过于简单，难以有效区分优秀设计师与普通设计师。这导致一些低质量的设计作品也能够进入平台进行交易，影响平台的整体质量。

（2）交易透明度不足：国内平台的交易过程中，很多平台对于买卖双方的信用体系和评价机制建设较为薄弱，用户往往缺乏足够的交易保障，容易产生交易纠纷。

1. 研究内容

对在线画稿交易平台的设计与实现进行研究，旨在构建一个高效流畅、安全稳定的线上艺术版权交易环境。这样的研究对于应对日趋激烈的市场竞争、满足消费者多样化需求、推动艺术品市场的健康发展具有重要意义。

本文的研究工作主要有以下几部分：首先，本文分别从平台运营模式、用户需求及市场趋势等方面进行探讨，明确了本研究的目的和意义。其次，深入探讨了在线画稿交易平台的系统设计，涵盖平台架构、主要功能模块及其实现的技术方案。在这部分，着重关注了设计师身份审核机制、用户余额充值、画稿交易等核心功能的设计思路及其实现过程。再次，通过对设计好的平台进行实际测试，评估其在用户加载时间、数据稳定性及安全性等多方面的表现，确保系统符合实际业务需求。

目前，在线艺术交易市场正处于快速发展阶段，对新兴技术的应用仍有待深入研究。本文通过对在线画稿交易平台的设计与实现，尝试为未来的研究提供理论与实践上的支持，促进艺术市场的进一步发展。总而言之，设计与实现在线画稿交易平台不仅是对现有技术的综合运用与创新，也是对电子商务领域中艺术作品交易途径的拓展。通过此研究，我们将能够为创作者提供一个更广阔的舞台，为消费者带来更丰富的艺术体验，同时为推动艺术作品市场向数字化、网络化发展趋势作出贡献。

1. 研究方法

本文的研究方法涵盖了用户需求分析、系统设计、开发实现以及系统测试等多个阶段。在用户需求分析阶段，采用了问卷调查以及访谈等方式收集潜在用户的反馈，这包括设计师和普通消费用户，目的在于准确地把握用户期望的功能。之后，根据这些初步数据对平台进行功能模块细化，重点关注用户交易过程中的便捷性和安全保障。[3]

在开发实现阶段，选定使用Python作为后端开发语言，鉴于其出色的数据处理能力和广泛的库支持。前端开发采用了Vue框架，为用户提供了平滑且直观的交互体验。数据库系统的选择上，采用MySQL来处理交易记录等结构化数据。

通过我们的研究与实践，旨在为在数字艺术品交易市场带来创新，同时也对相应的电子商务模式和用户行为做出了深入分析与探讨。在未来的工作中，预期可以将更多新兴技术，如人工智能和区块链，进一步整合到我们的平台中，从而为用户带来更全面、更加安全便捷的艺术交易体验。

# 开发技术介绍

1. Django

Django 是一个开源的Web后端框架，旨在帮助开发人员高效地构建 Web 后台。其核心特点包括：

1. 完备的功能库：Django 提供了众多内置功能，包括用户认证、权限管理、管理后台、表单处理、数据库模型等。这些功能帮助开发人员专注于业务逻辑的实现，而无需重复造轮子。
2. MVC架构（MTV模型）：Django 采用了 MTV 模型（即模型、模板、视图），它将业务逻辑和展示层分开，确保了代码的高内聚低耦合。
3. 自动化管理后台：Django 内置了一个强大的后台管理系统，开发人员可以通过这个后台管理界面轻松地对数据模型进行增、删、改、查等操作。它大大提高了开发效率，尤其是对于数据管理相关的需求。
4. 安全性：Django 提供了多种内建的安全措施，防止常见的安全漏洞，如 SQL 注入、跨站脚本攻击（XSS）和跨站请求伪造（CSRF）。

通过 Django 提供的功能和架构设计，本项目能够高效地实现各种核心功能，如用户管理、交易流程、画稿展示等，同时保障系统的稳定性和安全性。

1. MySQL

MySQL 是一个开源的关系型数据库管理系统，广泛应用于 Web 开发中。其特点包括：

1. 高效性和可靠性：MySQL 是一种高性能、可扩展的数据库管理系统，适用于处理大量数据。
2. ACID 支持：MySQL 遵循 ACID 特性（原子性、一致性、隔离性、持久性），保证了数据的完整性和事务的安全性。
3. 支持复杂查询：MySQL 支持多表联接、事务、索引等复杂查询，能够满足本平台在用户管理、画稿管理、交易管理等方面的需求。

本项目中的数据库设计基于 MySQL，存储用户信息、画稿信息、交易记录等核心数据。使用 Django 的 ORM（对象关系映射）框架，开发人员可以更加便捷地与数据库进行交互，无需编写大量 SQL 语句。

1. JSON Web Token（JWT）

JSON Web Token基于JSON格式的开放标准，用于在网络应用环境间传递声明。JWT的核心作用是在无需存储会话信息的情况下，安全地传递用户身份和权限信息，广泛应用于认证和授权机制中。JWT主要由头部、负载、签名三部分组成：

1. 头部：包括两部分信息，令牌的类型和所使用的签名算法（例如HMAC SHA256或RSA）。
2. 负载（Payload）：包含需要传递的声明（Claims），这些声明可以包括用户身份信息或其他元数据（如过期时间）。需要注意的是，这部分数据并未加密，仅是经过Base64编码，任何人都可以解码查看。
3. 签名（Signature）：通过使用头部指定的签名算法（如HMAC SHA256），对编码后的头部和负载进行加密，生成签名，从而确保JWT的完整性，防止数据被篡改。

在本系统中，JWT被用于用户的认证与授权。当用户登录时，系统通过验证用户名和密码生成JWT，前端在随后的API请求中携带此JWT，后端通过验证JWT的有效性来判断用户是否有权限访问相关资源。通过这种方式，JWT帮助提高了系统的安全性与性能，避免了服务器对会话信息的存储和管理。

1. Vue.js

Vue.js 作为前端框架，负责实现页面的动态交互。其核心特性，如双向数据绑定、组件化开发和虚拟Dom，使得开发过程更加的高效，用户体验更加的流畅。

具体而言，Vue 的组件化机制帮助拆分平台的各个功能模块，提高了代码的复用性和可维护性。同时，利用 Vue 的响应式数据系统，使得页面能够实时更新用户的操作反馈，增强交互体验。

此外，Vue.js 结合 Vue Router 和 Vuex，可以进一步优化平台的路由管理和状态管理，确保复杂业务逻辑的高效处理。通过这些技术的应用，前端系统能够在保证性能的同时，为用户提供直观、流畅的操作体验。

1. Ant Design Vue

Ant Design Vue 是基于 Ant Design 设计语言的 Vue 组件库，它提供了一套高质量的 UI 组件，能够帮助开发人员快速构建现代化的 Web 应用界面。Ant Design Vue 的优势包括：

1. 丰富的 UI 组件：提供了表单、按钮、输入框、表格等常用的 UI 组件，开发人员可以直接调用，不必从零开始开发。
2. 响应式设计：Ant Design Vue 支持响应式布局，能够在不同设备上呈现出优雅的界面，适应手机、平板和桌面端的需求。
3. 一致性设计：Ant Design 提供了统一的设计规范，使得整个系统的界面风格保持一致，提升用户体验。

在本项目中，Ant Design Vue 被用来构建平台的各类页面，如注册、登录、个人中心、画稿管理、交易页面等。通过它提供的 UI 组件，开发人员能够快速搭建出符合现代设计标准的用户界面。

# 系统分析

1. 市场分析
   1. 用户画像分析

在用户画像的构建过程中，我们使用了问卷调查、爬虫等方法，采集了用户的在线行为数据，包括他们的浏览习惯、偏好设置以及购买历史，旨在理解用户的具体需求，进而提供更为个性化的服务。我们综合了用户画像及消费行为，对潜在的参与者进行了深入分析。研究结果显示，该市场的主要用户群体分为三类：一是热爱艺术并积极参与收藏的个人用户，这类用户更关注作品的个性化和稀缺性；二是有商业使用需求的企业用户；三是专业艺术家和设计师。后二者不仅关注作品本身的优劣，还关注作品的版权合规和销售安全问题。针对这类特定的客户群，我们的平台在确保画作原创性的认证机制上投入了大量的技术研发工作，并对版权保护进行了严格的设计。

* 1. 竞争对手分析

在对竞争对手平台功能的比较研究中，我们发现目前市场上的同类平台多以画稿的展示和促进销售为主，但缺乏有效的用户交互机制。在此基础之上，我们分析了供需关系，在画稿市场中，由于作品的个性化和不可替代性，导致了一种特殊的供需关系。消费者对于作品的需求不仅来自于审美和情感的附加值，还可能受到市场稀缺性和作品热度的影响。因此，在线画稿交易平台在促成交易之前，需要建立一个交流互动性强的环境，简而言之，需增加点赞、评论等功能，进而根据交互数据，优化画稿展示。

* 1. 作品分析

数字艺术作品的色彩、完成度和技法语言对用户选择有着巨大的影响。数据显示，用户偏好色彩明朗、表现力强、完成度高的作品，并且在评估作品时，技法的熟练程度是他们关注的重要因素。这一结果对设计在线画稿交易平台的展示、分级以及定价监管等功能具有启示作用，作者在创作时应考虑到这些因素，平台在功能设计上也需要潜移默化地倡导作者创作这类偏向的作品。

1. 功能需求分析
   1. 需求概述

本系统是一个基于 Web 的在线画稿交易平台，主要面向设计师和普通用户。普通用户可以浏览和购买画稿，同时可以申请成为设计师，在管理员审核通过后拥有上传画稿的权限。平台支持画稿管理、交易功能、用户管理等。

* 1. 角色划分

本系统的用户分为以下三种角色：

1. 普通用户：可浏览画稿、搜索画稿、购买画稿，申请成为设计师。
2. 设计师：可以上传、编辑和管理自己的画稿。
3. 管理员：负责审核设计师申请、画稿审核、交易管理等。
   1. 主要功能模块
4. 用户管理模块

* 用户注册与登录：支持普通用户账号注册和身份验证。
* 个人信息管理：用户可以修改个人基本信息，如昵称、头像、密码等。
* 身份认证：普通用户可以申请成为设计师，需提交相关资料，由管理员审核。
* 钱包管理：用户可以查看余额、充值和提现。
* 浏览记录：系统记录用户的浏览历史，用户可查看自己的浏览记录。

1. 画稿管理模块

* 购买画稿：普通用户可支付购买画稿，购买成功后可下载高清、无水印版本。
* 订单管理：用户可查看已购买的画稿订单，设计师可查看自己出售的画稿订单。
* 支付功能：支持在线支付，具体支付方式待定。
* 交易状态管理：订单状态包括“待支付”、“已支付”、“已完成”、“已取消”等。

1. 订单管理模块

* 创建订单：用户可通过购买画稿创建一个订单
* 搜索订单：通过关键字，时间，金额等条件查询订单
* 取消订单：用户提交取消申请，经管理员审批后，可以退款
* 删除订单：将已有的订单删除

1. 可行性分析
   1. 技术可行性

本系统采用 Django 作为后端框架，提供稳定可靠的 API 服务，同时使用 MySQL 作为数据库，保障数据的高效存储与管理。前端采用 Vue.js 结合 Ant Design 组件库，实现良好的用户交互体验。此外，采用 JWT 进行用户身份验证，提高安全性。整体架构基于成熟的 Web 技术，具有良好的可扩展性和可维护性。

* 1. 经济可行性

本系统的开发成本主要包括服务器租赁、域名注册以及支付接口的集成费用。初期可以选择云服务器，以降低成本，并根据用户增长情况逐步扩展服务器资源。此外，平台可以通过收取交易佣金、广告投放等方式获取收入，从而维持运营并实现盈利。

* 1. 用户可行性

本系统的目标用户主要是设计师和普通用户。设计师希望通过平台出售作品获利，而普通用户希望获得高质量的设计资源。因此，系统提供便捷的交易流程、完善的售后支持以及丰富的画稿资源，以吸引用户使用。此外，通过推广和市场营销，提高平台的知名度和用户粘性。

* 1. 运营可行性

平台的运营涉及用户引流、画稿审核、交易管理等方面。初期可通过社交媒体、设计师合作等方式吸引用户入驻，同时制定合理的审核机制，确保画稿质量。通过数据分析优化平台运营策略，提高用户活跃度和交易量，保障平台的长期稳定发展。

# 系统设计

4.1系统架构设计

4.1.1 总体架构

本系统采用了分层架构的设计模式，整体分为四个主要部分：操作系统层、数据库层、Web应用服务层和用户层。每一层都承担着特定的职责，确保系统在高效、可靠的基础上运行，并具备良好的可扩展性和易维护性，如图4-1所示：

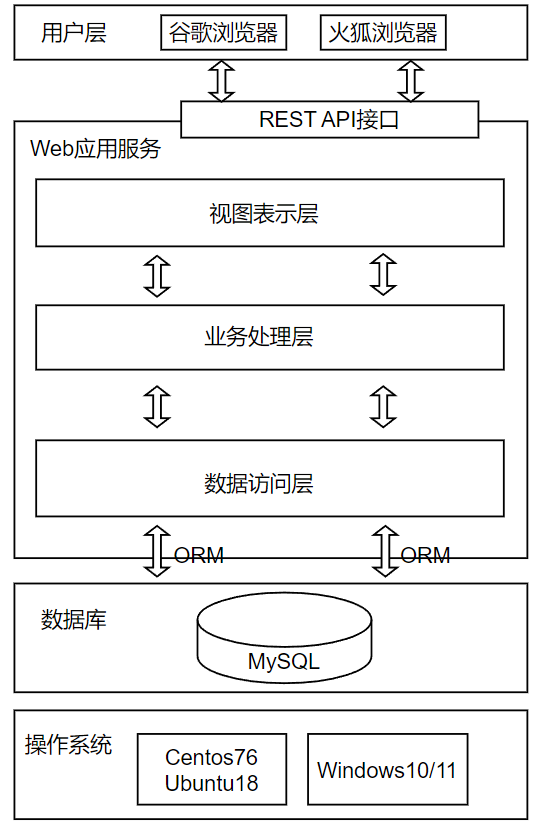


图4-1 系统架构总体设计

（1）操作系统层

操作系统作为整个系统的基础支撑，负责管理所有的硬件资源和软件资源，是上层系统的基石。操作系统层的稳定性会直接影响上层系统，因此操作系统必须具备超强的稳定性和高性能。

（2）数据库层

数据库层作为数据的载体，包括用户信息、交易记录、设计稿件、订单信息等。该层采用MySQL作为数据库系统，MySQL是高性能并且稳定的关系型数据库，能够高效存储和管理数据，并支持复杂的查询操作。在数据库层，采用合理的数据库设计，确保数据的完整性、一致性、隔离性和持久性（ACID特性）。通过对数据进行索引和优化，确保系统在数据量增长时依然能够保持较高的查询效率。

（3）Web应用服务层

Web应用服务层是本次开发的主要阵地，也是整个后台系统的核心，负责处理所有的业务逻辑，并通过REST API与用户层进行交互。Web应用服务层分为三个子层：

1. 视图表示层：这一层负责处理用户请求并返回响应，主要通过REST API接口暴露服务。它专门接收来自前端的请求，调用业务处理层进行处理，当拿到结果后再返回给用户，其强大的路由系统和请求处理机制可以保证请求的高效处理。
2. 业务处理层：这一层涵盖了系统的核心业务逻辑，如用户注册、设计师审核、交易管理等。它通过调用数据访问层获取或更新数据库中的数据，并按照一定的业务规则进行处理。业务处理层的设计要注重解耦与可复用性，确保系统能够随着需求变化而灵活扩展。
3. 数据访问层：负责跟数据库进行交互，提供对数据的增、删、改、查等操作。通过使用Django ORM（对象关系映射）框架，数据访问层能够大大地简化数据库操作，从而提高开发效率。同时，数据库层的优化、索引设计等也会影响到这一层的性能。

（4）用户层

用户层是系统的最外层，用户直接与其交互。前端采用Vue.js + Ant Design Vue + Router+Store+axios构建。通过与Web应用服务层暴露的REST API接口进行数据交互，实现各类功能，如用户注册、设计稿查看、交易买卖等。

在用户层中，不同角色（普通用户、设计师、管理员）会根据各自的权限，看到不同的功能模块和界面。前端通过Vuex管理状态，使用Vue Router进行路由控制，并通过API获取后端数据，实现动态交互。

4.1.2 逻辑架构设计

为了支持在线画稿交易平台的高效运作，系统被划分为多个功能模块，每个模块都有其特定的目的和职责。系统逻辑架构图如图4-2所示

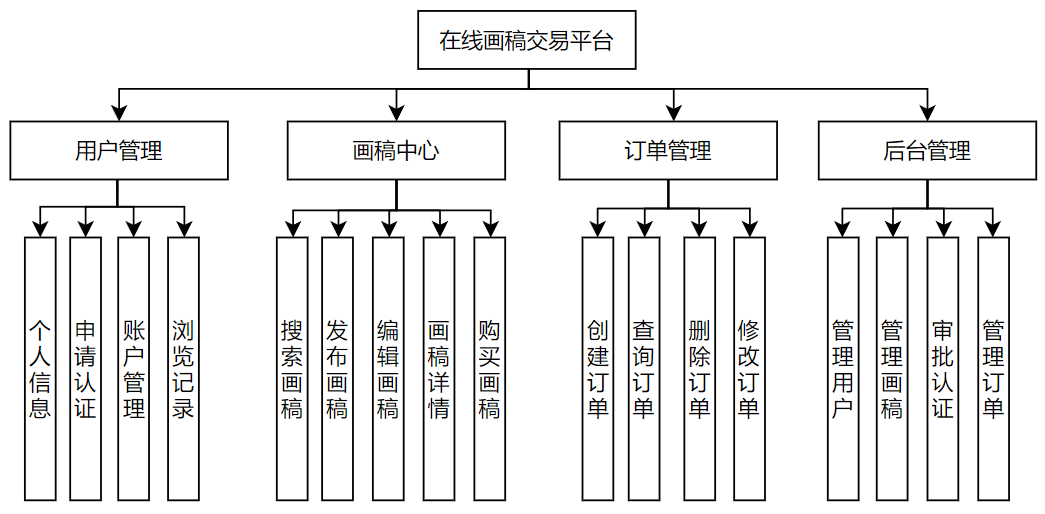


图4-2 逻辑架构图

在用户管理模块中，系统为用户提供了一个全面的个人信息管理系统，使他们能够轻松地查看并更新自己的基本信息。此外，该模块还支持艺术家进行实名认证申请，通过提交相关资料提升个人作品的可信度，同时提供账户管理功能，包括密码修改及邮箱或手机号绑定等操作，确保用户信息的安全与隐私。

画稿中心模块作为平台的核心部分，提供了丰富的画稿浏览、发布、编辑以及购买功能。用户可以根据关键词搜索自己感兴趣的画稿，而艺术家则可以上传他们的作品，并设置价格和版权信息。对于已发布的画稿，设计师有权对其进行编辑调整，如价格修改或描述补充等。当用户找到心仪的作品时，可以通过详细的画稿详情页面了解更多信息，然后直接进入购买流程完成交易。

订单管理模块负责处理从创建到完成的整个购物流程。一旦用户决定购买某幅画稿，系统将自动生成订单。用户可以方便地查看所有订单的状态，包括待支付、已支付等。此外，未支付的订单可以被删除或者在一定条件下进行修改，例如更改收货地址，确保交易的灵活性和准确性。

后台管理模块专为管理员设计，用于对平台进行全面监控和维护。管理员可以审核新上传的画稿，确保内容合法且质量上乘；审批用户的实名认证请求；以及处理任何异常订单问题。这个模块保证了平台的规则得以执行，同时也保护了用户的权益。

4.2分区模块设计

为了保证各个模块之间的低耦合和高内聚，系统将按照功能进行分区，每个功能区作为一个独立模块进行开发。以下是各模块的具体设计内容。

4.2.1 用户管理模块

用户管理模块负责用户的注册、登录、身份认证以及角色管理。系统根据不同角色提供不同的功能和权限，包括普通用户和设计师角色的审核、权限分配。

功能：

* 普通用户和设计师的注册与登录
* 设计师申请审核机制
* 用户角色的动态管理（普通用户/设计师）

数据存储：

* users表：保存用户的基本信息、登录凭证和角色信息

4.2.2 画稿管理模块

画稿管理模块是系统的核心部分，设计师通过该模块上传、修改和删除自己的画稿。普通用户可以查看画稿，进行搜索、浏览，最终完成交易。

功能：

* 设计师上传、修改和删除画稿
* 审核机制，管理员审核画稿的合法性
* 画稿展示与搜索功能，用户可查看、筛选和搜索画稿

数据存储：

* draft表：记录画稿的标题、价格、类别、描述、设计师信息、状态（待审核/已发布）
* category表：画稿的分类信息，如插画、设计稿等

4.2.3 订单管理模块

订单管理模块是平台中处理用户购买画稿和交易的关键模块，负责管理用户下单、订单支付、订单状态更新及退款等操作。此模块保证交易的顺利进行并记录每个订单的详细信息。

功能：

* 下单：普通用户选定并购买画稿时，后台会生成订单，记录商品、用户以及设计师的相关信息。
* 支付：订单生成后，用户可以到达支付页面，点击支付按钮，系统会确认支付成功后更新订单状态。
* 订单状态管理：订单状态可以划分为待支付、已支付、待发货、已发货、已完成等。平台根据支付情况和用户操作进行订单状态的变更。
* 订单查询与管理：用户仅能查看和取消自己的订单，管理员可以查看并管理这个平台所有的订单信息。

数据存储：

* order表：存储画稿订单的基本信息，主要字段为订单号、买家、卖家、画稿信息、订单状态、创建时间等。

**订单的状态可以从下单开始经过一系列步骤，主要包括：**

* 待支付：用户提交订单后，系统生成订单并设置为待支付状态。
* 已支付：用户完成支付后，订单进入已支付状态，系统更新支付信息并准备发货。
* 已发货：当设计师或管理员发货后，订单状态更新为已发货，用户可以查看发货信息。
* 已完成：订单完成交易，用户确认收货后，状态更新为已完成，订单结束。

4.2.4 后台管理模块

管理员后台模块是平台管理的核心，负责审核用户和画稿、管理系统设置等。管理员可以对平台上的所有用户和交易进行操作和控制。

功能：

* 用户管理，包括审核设计师申请、禁用用户等
* 画稿管理，管理员可以审核设计师发布的画稿，确保其符合平台规定；此外，管理员有权删除违规画稿，避免不符合规范的内容影响平台的正常运营。
* 订单管理，在交易过程中，如果出现纠纷或需要干预的情况，管理员可以介入订单，进行处理和调整。管理员可以查看订单的详细信息，确认是否存在问题，并进行退款或其他操作，保障平台的公正性和用户的权益。
* 系统设置，如权限配置等

4.3交互模式设计

交互模式设计涵盖用户在平台上的所有交互行为，包括浏览、搜索、购买、发布画稿、申请设计师身份等。一个直观且友好的交互模式，能够大大提升用户体验，帮助平台更好地服务用户。

4.3.1 用户角色与操作流程

主要有三种用户角色：管理员、设计师和普通用户，每种角色的操作流程和交互方式有所不同。

**普通用户**：

* 浏览画稿：普通用户进入平台后，可以查看画稿的浏览列表，并通过分类、筛选、排序等方式快速找到感兴趣的画稿。点击某个画稿后，用户进入画稿详情页，查看画稿的大图、设计师信息以及价格等。
* 搜索功能：普通用户可以使用关键词搜索画稿，支持按标题、标签、价格等条件进行搜索，快速定位到相关画稿。
* 购买画稿：普通用户点击“购买”按钮后，系统会生成订单，用户填写收货信息并选择支付方式。支付完成后，订单状态更新为“已支付”，并通知设计师发货。
* 申请设计师身份：普通用户可以通过平台提供的申请按钮，提交设计师身份申请。申请提交后，管理员会进行审核，审核通过后，普通用户变为设计师。

普通用户用例如图4-3所示：

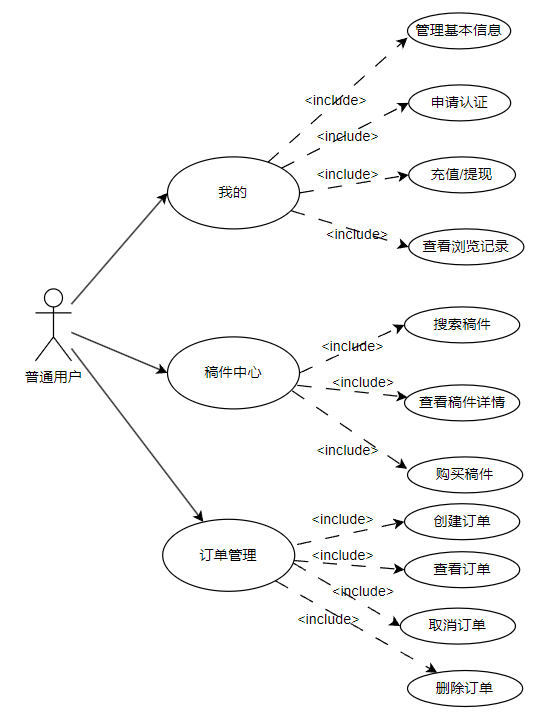


图4-3 普通用户用例图

**设计师：**

* 发布画稿：设计师可以通过“发布画稿”按钮进入画稿上传页面，填写画稿标题、描述、价格、分类等信息，上传画稿文件，并发布到平台上供其他用户浏览和购买。
* 管理画稿：设计师可以管理自己发布的画稿，查看每个画稿的购买情况，修改画稿的价格或描述，甚至删除不再展示的画稿。
* 查看订单：设计师可以查看自己的订单，确认用户支付情况。

设计师用例如图4-4所示：

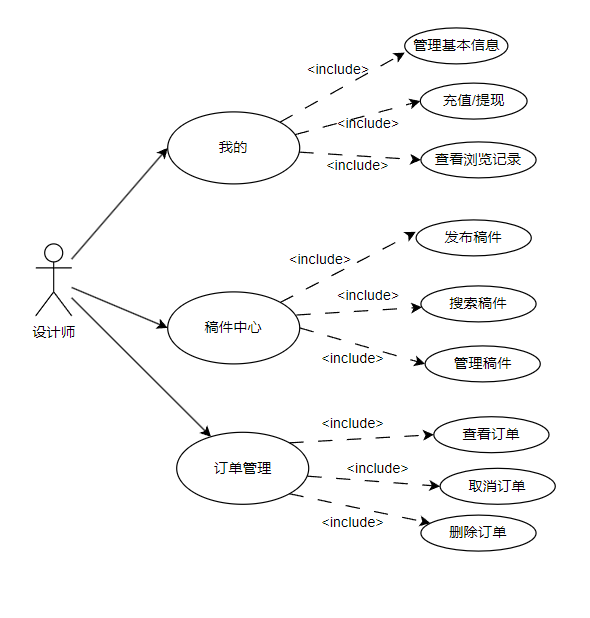


图4-4 设计师用例图

**管理员**：

* **管理用户**：管理员可以查看所有用户信息，审核普通用户提交的设计师身份申请，批准或拒绝申请。
* **管理画稿**：管理员可以查看平台上的所有画稿，审核画稿的合法性、质量等，并根据规则删除违规内容。
* **管理订单**：管理员可以查看所有的订单记录，处理退款和纠纷等问题，确保交易的顺利完成。

管理员用例如图4-5所示：

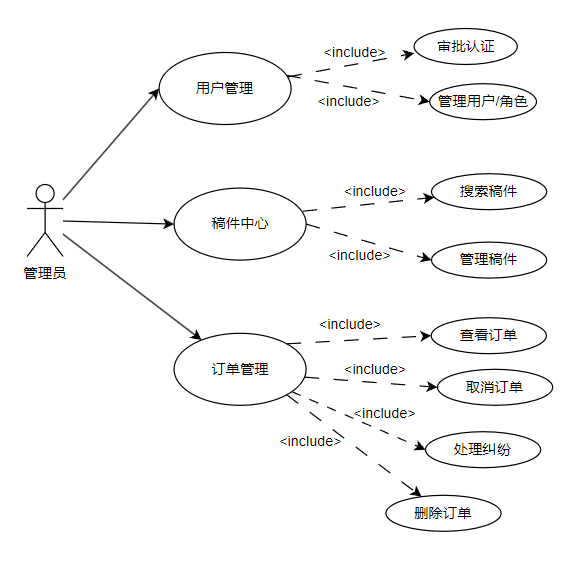
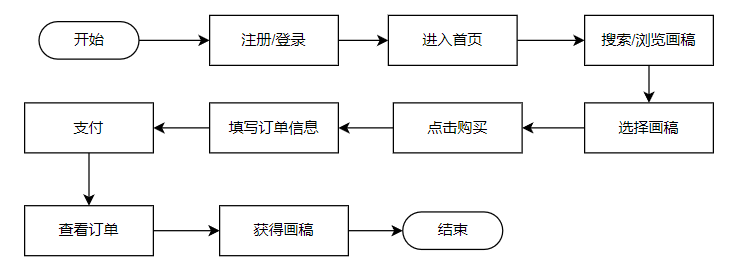


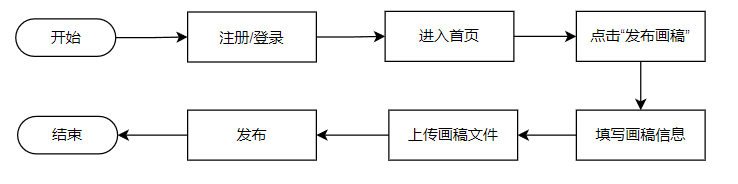
图4-5管理员用例图

4.3.2 交互流程设计

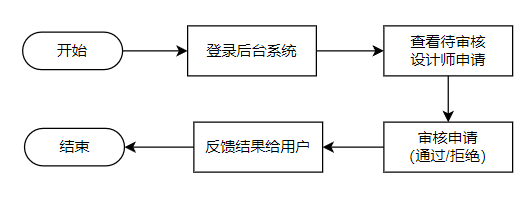
（1）普通用户注册与购买画稿流程



（2）设计师发布画稿流程



（3）管理员审核设计师申请流程



4.4数据库设计

4.4.1 E-R图

系统数据模型由4个实体组成，分别是用户、画稿、订单、画稿类别。实体之间的关系如图4-6所示：

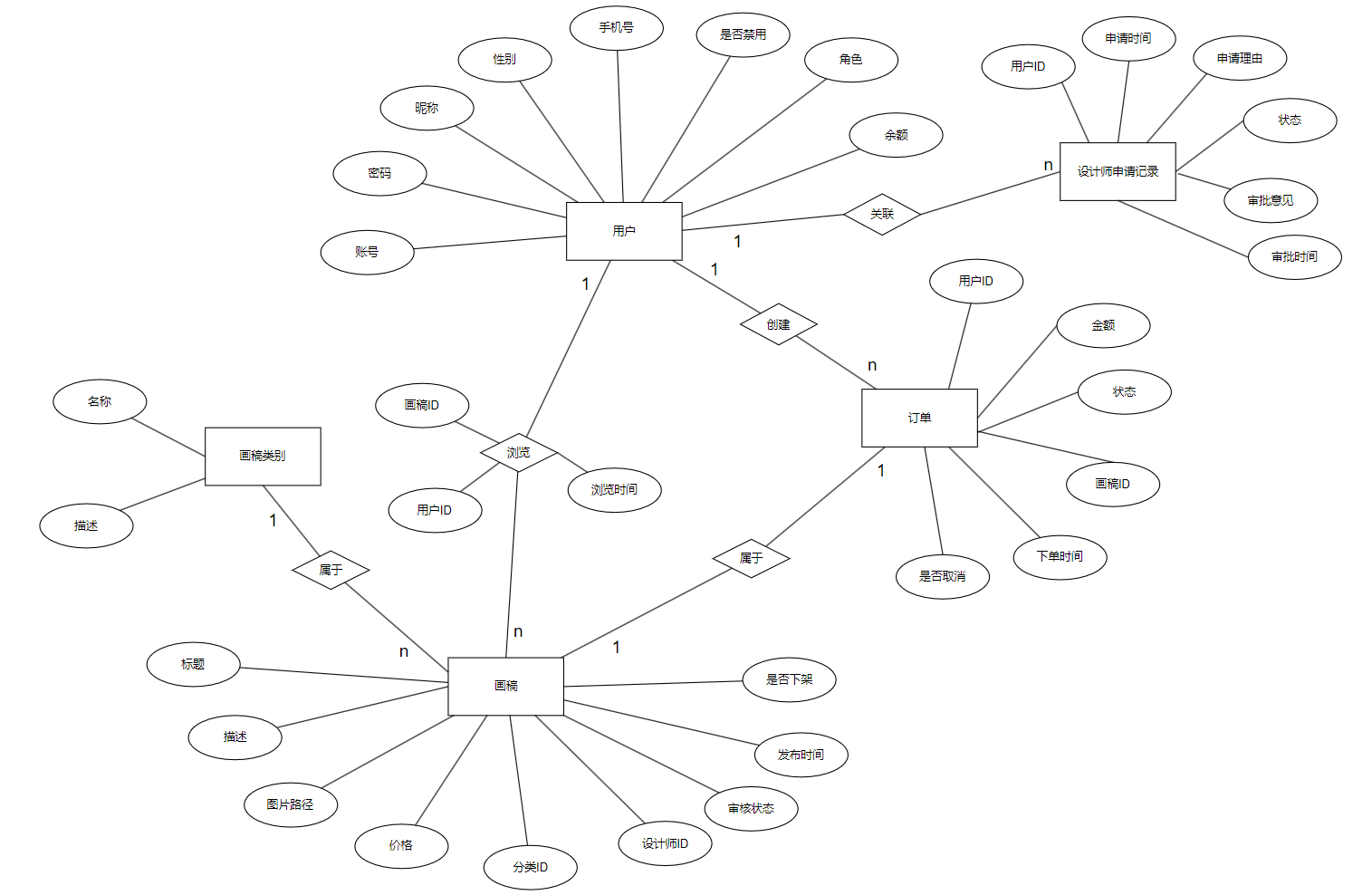


图4-6 实体-关系图

4.4.2 数据表设计

用户表用于存储系统用户的基本信息，包括用户ID作为主键并自增，账号用于唯一标识用户，密码用于身份验证，昵称记录用户的个性化名称。表中还包含性别、手机号等基本信息。用户的角色字段用于区分不同权限，余额用于账户资金管理，是否禁用字段用于控制用户访问权限。该表确保了系统的用户管理功能，见表4-1。

表4-1 用户表（user）

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 字段类型 | 字段描述 | 允许为空 | 主键 | 外键 | 自增 |
| 1 | id | bigint(20) | 用户ID | 否 | √ |  | √ |
| 2 | username | varchar(64) | 账号 | 否 |  |  |  |
| 3 | password | varchar(1024) | 密码 | 否 |  |  |  |
| 4 | nickname | varchar(32) | 昵称 | 否 |  |  |  |
| 5 | gender | int | 性别 | 否 |  |  |  |
| 6 | phone | varchar(32) | 手机号 | 否 |  |  |  |
| 7 | is\_ban | int | 是否禁用 | 否 |  |  |  |
| 8 | role | varchar(20) | 角色 | 否 |  |  |  |
| 9 | balance | float | 余额 | 否 |  |  |  |

画稿表用于存储平台上的设计作品信息，每个画稿都有唯一的画稿ID作为主键。标题和描述用于展示画稿的基本信息，图片存储路径记录画稿的具体存放位置。价格字段表示画稿的标价，类别ID用于分类管理，设计师ID关联发布画稿的用户。审核状态用于标识画稿的审批进度，发布时间记录画稿上线时间，是否下架字段用于控制画稿的可见性，见表4-2。

表4-2 画稿表（draft）

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 字段类型 | 字段描述 | 允许为空 | 主键 | 外键 | 自增 |
| 1 | id | bigint(20) | 画稿ID | 否 | √ |  | √ |
| 2 | title | varchar(64) | 标题 | 否 |  |  |  |
| 3 | description | varchar(1024) | 描述 | 否 |  |  |  |
| 4 | image\_url | varchar(2048) | 图片存储路径 | 否 |  |  |  |
| 5 | price | float | 价格 | 否 |  |  |  |
| 6 | category | bigint | 类别ID | 否 |  | √ |  |
| 7 | user\_id | bigint | 设计师ID | 否 |  | √ |  |
| 8 | status | varchar(20) | 审核状态 | 否 |  |  |  |
| 9 | online\_time | bigint | 发布时间 | 否 |  |  |  |
| 10 | is\_outline | int | 是否下架 | 否 |  |  |  |

浏览记录表用于记录用户对画稿的浏览情况。每条浏览记录都有唯一的ID作为主键，设计师ID用于标识浏览用户，画稿ID用于关联被浏览的作品。该表能够帮助平台分析用户的兴趣偏好，为个性化推荐提供数据支持，见表4-3。

表4-3 浏览记录表(browse)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 字段类型 | 字段描述 | 允许为空 | 主键 | 外键 | 自增 |
| 1 | id | bigint(20) | ID | 否 | √ |  | √ |
| 2 | user\_id | bigint | 设计师ID | 否 |  | √ |  |
| 3 | draft\_id | bigint | 画稿ID | 否 |  | √ |  |

订单表用于记录用户购买画稿的交易信息。每个订单由唯一的订单ID标识，包含用户ID、画稿ID、交易金额等关键信息。订单状态用于跟踪交易进度，而下单时间记录订单创建时间。是否取消字段用于标识订单是否被取消。该表为平台的交易管理提供了基础数据支持，见表4-4。

表4-4 订单表（order）

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 字段类型 | 字段描述 | 允许为空 | 主键 | 外键 | 自增 |
| 1 | id | bigint(20) | 订单ID | 否 | √ |  | √ |
| 2 | user\_id | bigint | 用户ID | 否 |  | √ |  |
| 3 | amount | float | 金额 | 否 |  |  |  |
| 4 | status | varchar(20) | 状态 | 否 |  |  |  |
| 5 | draft\_id | bigint | 画稿ID | 否 |  |  |  |
| 6 | create\_time | bigint | 下单时间 | 否 |  | √ |  |
| 7 | is\_cancel | int | 是否取消 | 否 |  |  |  |

设计师申请记录表用于存储用户申请成为设计师的相关信息。该表包含用户ID、申请理由、审批状态等字段，以便平台管理员审核用户申请。审批意见字段记录审核结果，审批时间和申请时间字段用于跟踪申请流程。该表有助于管理设计师的入驻流程，确保平台设计师的质量和合规性，见表4-5。

表4-5 设计师申请记录表（designer\_application\_record）

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段名称 | 字段类型 | 字段描述 | 允许为空 | 主键 | 外键 | 自增 |
| id | bigint(20) | 订单ID | 否 | √ |  | √ |
| user\_id | bigint | 用户ID | 否 |  | √ |  |
| reason | varchar(256) | 申请理由 | 否 |  |  |  |
| status | varchar(20) | 状态 | 否 |  |  |  |
| approval\_opinions | varchar(256) | 审批意见 | 否 |  |  |  |
| approval\_time | bigint | 审批时间 | 否 |  | √ |  |
| create\_time | bigint | 申请时间 | 否 |  |  |  |

# 系统实现

5.1 开发环境搭建

本系统在最终在Linux服务器上运行。在开发过程中，开发工具与环境配置如下：

* 操作系统：Windows11专业版
* IDE：Pycharm 2024.3
* Python解释器：Python3.11
* 前端开发环境：Node v18
* 数据库：MySQL 8.0

5.2 注册/登录（JWT）

在本系统中，用户的注册和登录功能通过 Django Rest Framework (DRF) 与 JWT (JSON Web Token) 结合实现。使用 JWT 实现用户认证和授权，能够有效保障用户数据的安全，并使得前后端的交互更加高效。

5.2.1 注册功能

用户注册时，系统会接收用户提交的用户名、密码、昵称等信息，并进行相应的校验。用户密码采用加密存储，以确保用户数据的安全性。密码的加密存储使用了 Django 自带的 make\_password 和 check\_password 方法。注册时需填写用户名、昵称、手机号、密码、确认密码、性别等信息，注册页面如图5-1所示：



图5-1 注册登录页面

**注册流程：**

（1）用户通过 POST 请求发送注册信息，包括用户名、密码、昵称等。

（2）后端校验用户提交的数据是否合法（如密码是否符合规则、用户名是否已被占用等）。

（3）密码采用加密方式存储，用户信息保存在数据库中。

（4）注册成功后，返回一个成功响应。

5.2.2 登录功能

登录功能通过用户名和密码验证用户身份。登录后，系统会生成 JWT token，返回给用户。JWT 是一种轻量级的认证方式，客户端每次请求时都可以携带该 token 来证明身份。登录页面如图5-2所示：

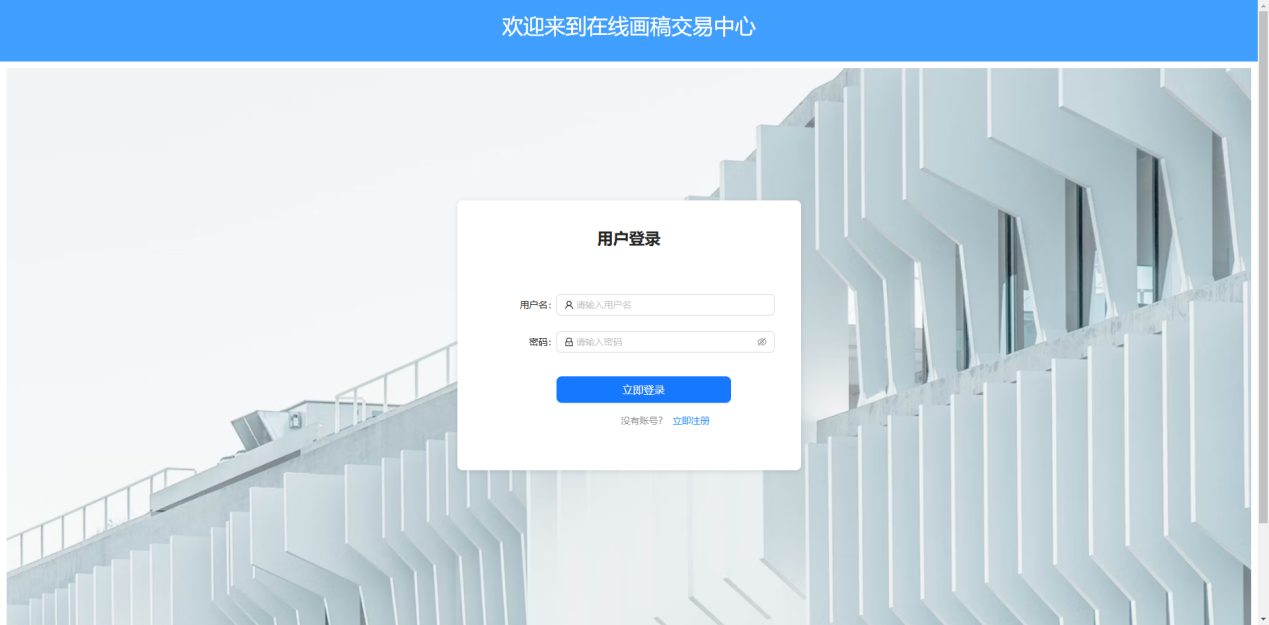


图5-2 登录页面

**登录流程：**

（1）用户通过 POST 请求发送用户名和密码。

（2）后端验证用户名和密码是否正确。

（3）如果验证通过，系统会生成 JWT token，并将其返回给用户。

（4）用户在后续请求中可以携带该 token 进行身份验证。

5.2.3 密码存储

在用户注册和密码校验过程中，密码从未以明文形式存储或传输。Django 提供了 make\_password 和 check\_password 方法，前者用于加密存储用户密码，后者用于在登录时验证密码是否正确。

**加密存储：**使用 make\_password 方法对用户密码进行加密。

**密码校验：**在用户登录时，使用 check\_password 校验用户输入的密码是否与存储的加密密码一致。

5.3 余额充值

在本系统中，用户可以通过支付宝进行余额充值。用户进入个人中心，点击立即充值，选择金额，确认后即可跳转至支付宝收款页面，扫码完成支付（同时支持账密登录）。图5-3是充值页面，图5-4是支付页面。

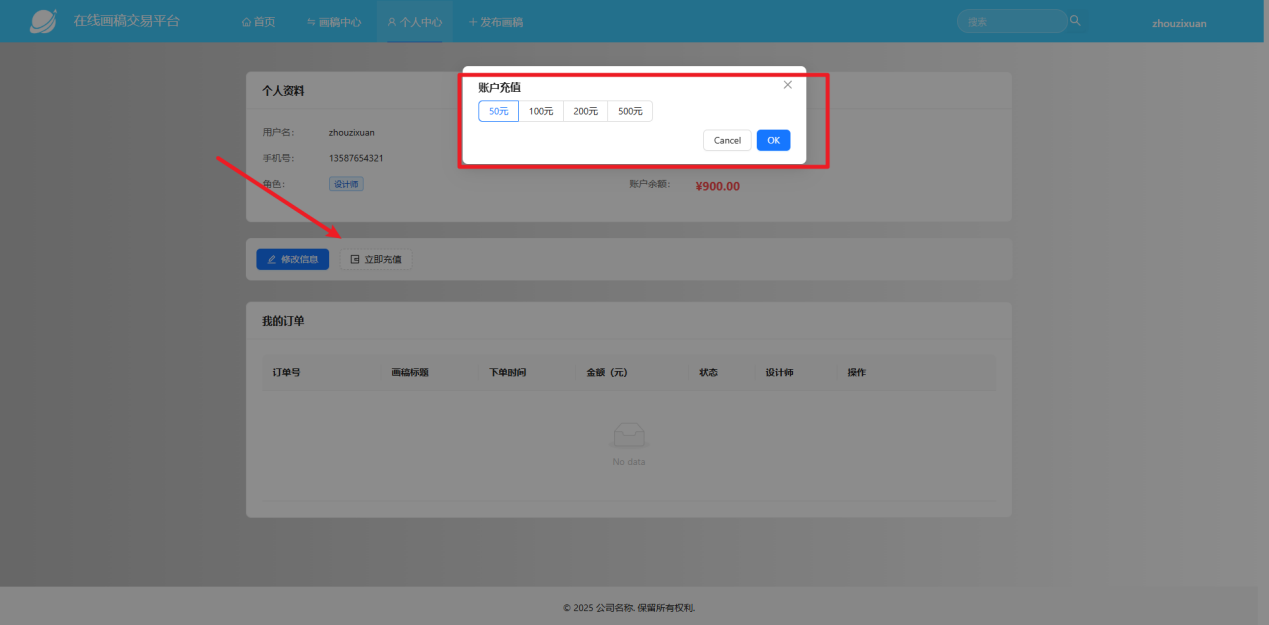


图5-4 充值页面

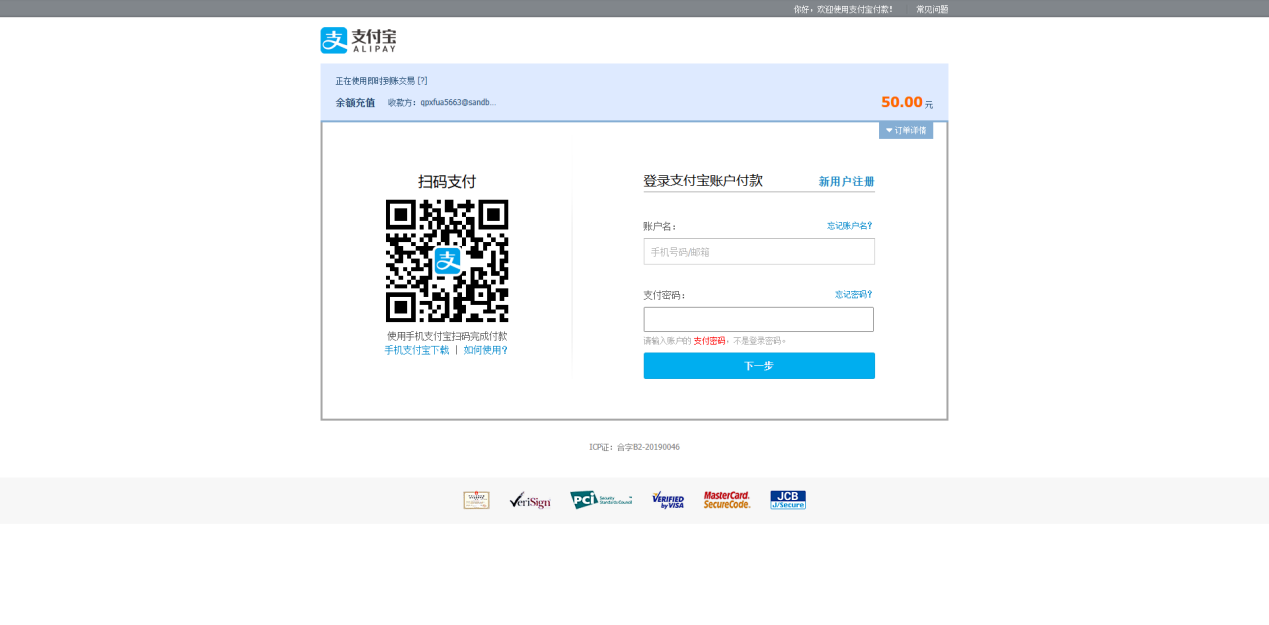


图5-5 支付页面

支付宝支付接入流程大致分为以下几个步骤：

（1）申请支付宝商户账号和支付宝支付接口

（2）安装支付宝 SDK

（3）生成订单并请求支付宝接口

（4）支付回调通知

（5）完成支付后更新用户余额

5.3.1 申请支付宝商户账号

进入支付宝开放平台，注册商户并创建应用，获得 app\_id 和 支付宝公钥、支付宝私钥，这些都是后续操作所必需的。

5.3.2 安装支付宝SDK并配置

在终端执行，pip install alipay-sdk-python，完成SDK安装，在项目的settings.py文件中，配置支付宝所需的配置信息：

ALIPAY\_CONFIG = {

'app\_id': '申请的app\_id',

'app\_private\_key': '申请的私钥',

'alipay\_public\_key': '申请的公钥',

'notify\_url': 'http://draft.com/alipay/notify', # 支付宝异步通知地址（假设系统域名为draft.com）

'return\_url': 'http://draft.com/alipay/return', # 支付宝同步通知地址（假设系统域名为draft.com）

}

5.3.3 支付回调通知

支付完成后，支付宝会向 notify\_url 提交异步通知，告诉系统支付结果。系统需要处理这些通知并根据通知内容更新用户余额。关键代码如下：

def async\_notify(self, trade\_status, out\_trade\_no, received\_amount,

trade\_no, seller\_id, buyer\_id):

"""

处理异步通知

:param trade\_status: 交易状态

:param out\_trade\_no: 订单号

:param received\_amount: 收到金额

:param trade\_no: 支付宝交易号

:return:

"""

order\_status = WalletOrderStatus.PAY\_FAILED.value

if trade\_status == "TRADE\_SUCCESS":

logger.info("支付成功，订单号:{}, 支付宝交易号:{}, 收到金额:{}".format(

out\_trade\_no, received\_amount, trade\_no))

order\_status = WalletOrderStatus.PAID.value

else:

logger.error("支付失败，状态：{}".format(trade\_status))

# 更新订单信息

wallet\_order = WalletOrder.objects.filter(order\_uuid=out\_trade\_no)

if not wallet\_order.exists():

raise DataNotExistsException("订单{}不存在".format(out\_trade\_no))

wallet\_order = wallet\_order.first()

wallet\_order.status = order\_status

wallet\_order.received\_time = int(time.time())

wallet\_order.seller\_id = seller\_id

wallet\_order.buyer\_id = buyer\_id

wallet\_order.save()

logger.info("订单{}状态已更新为：{}".format(out\_trade\_no, order\_status))

# 新增个人余额

user\_id = wallet\_order.user\_id

user = User.objects.get(id=user\_id)

user.balance += float(received\_amount)

user.save()

logger.info("用户{}余额已更新为：{}".format(user\_id, user.balance))

该视图处理支付宝支付回调通知，验证支付结果并更新用户余额。

5.4 设计师认证/审核机制

在本系统中，用户注册后默认是普通用户。为了确保设计师的质量和平台的规范性，普通用户需要申请成为设计师，并通过管理员审核才能获得设计师身份。在审核机制中，设计师申请包含审核通过、审核拒绝和待审核三种状态。通过这种方式，平台能够对设计师进行有效的管理与控制。流程如图5-6所示：

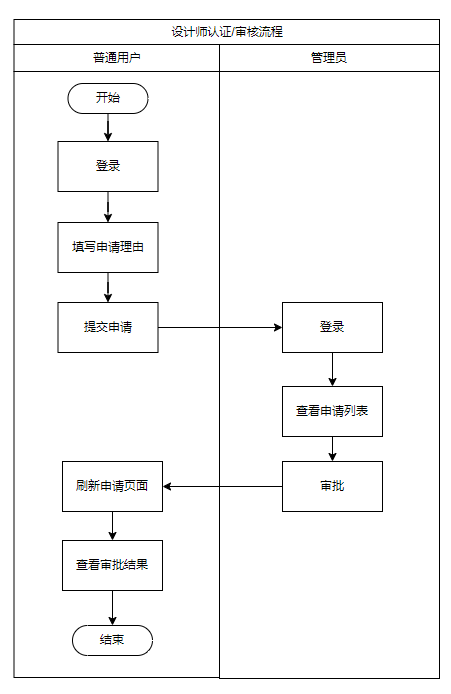


图5-6 设计师认证/审批泳道图

5.4.1 申请成为设计师

用户可以在系统中发起设计师申请，提交成为设计师的理由。管理员会对每个申请进行审核，并根据审核结果决定是否批准该用户成为设计师。设计师申请涉及到的核心字段包括申请理由、审核状态、拒绝原因、申请时间和审批时间等。申请成为设计师页面如图5-7所示：

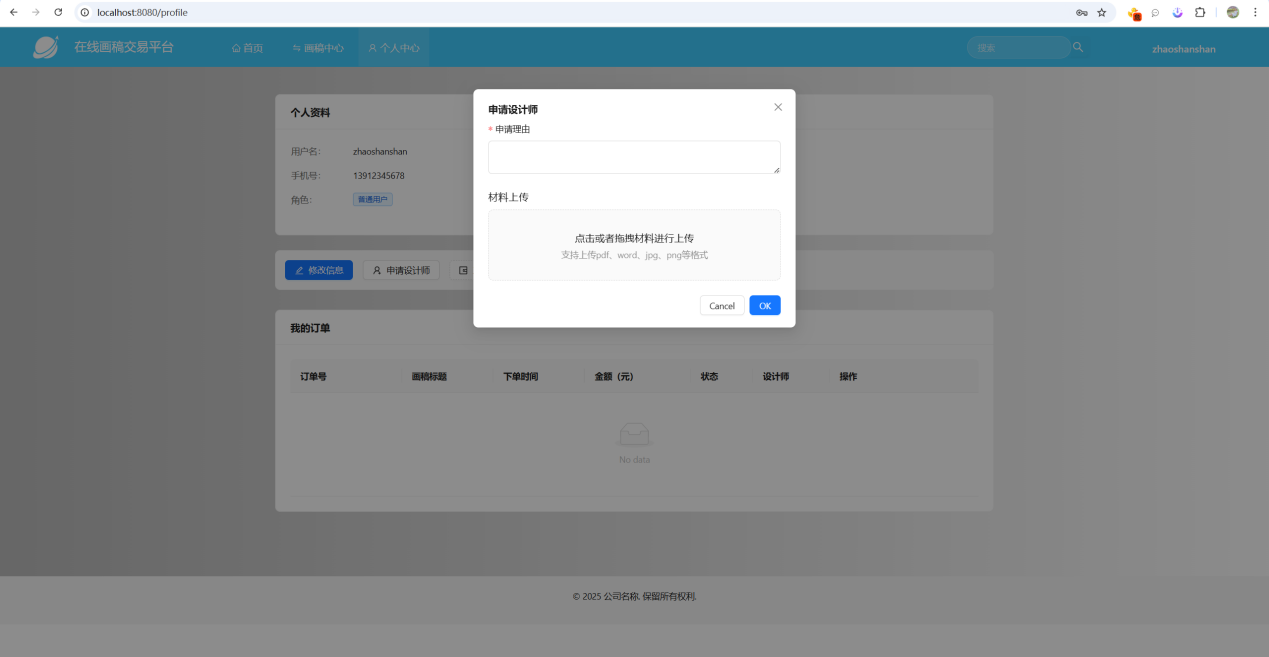


图5-7 申请设计师页面

5.4.2 管理员审核设计师申请

管理员通过后台接口对设计师申请进行审核。管理员可以选择通过或拒绝用户的设计师申请。通过审核后，用户的角色将更新为“设计师”，并且 DesignerApplication 表中的状态将被更新为“已通过”。如果申请被拒绝，拒绝理由也会被记录。管理员审批页面如图5-8所示：

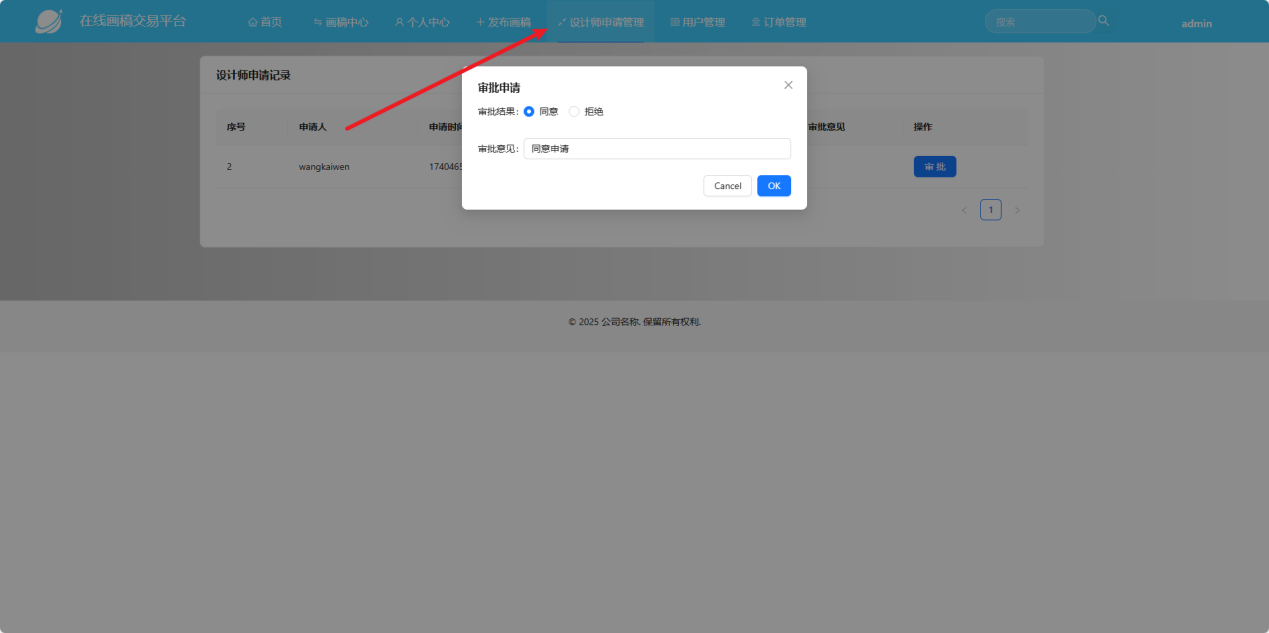


图5-8 管理员审批申请页面

5.4.3 用户查询申请状态

用户可以查询自己的设计师申请状态。系统会根据 DesignerApplication 表中的状态字段返回相应的结果。如果用户的申请处于“待审核”状态，系统会提示用户等待审核；如果申请已经通过或被拒绝，系统会返回审核结果。用户查看审批结果如图5-9所示：

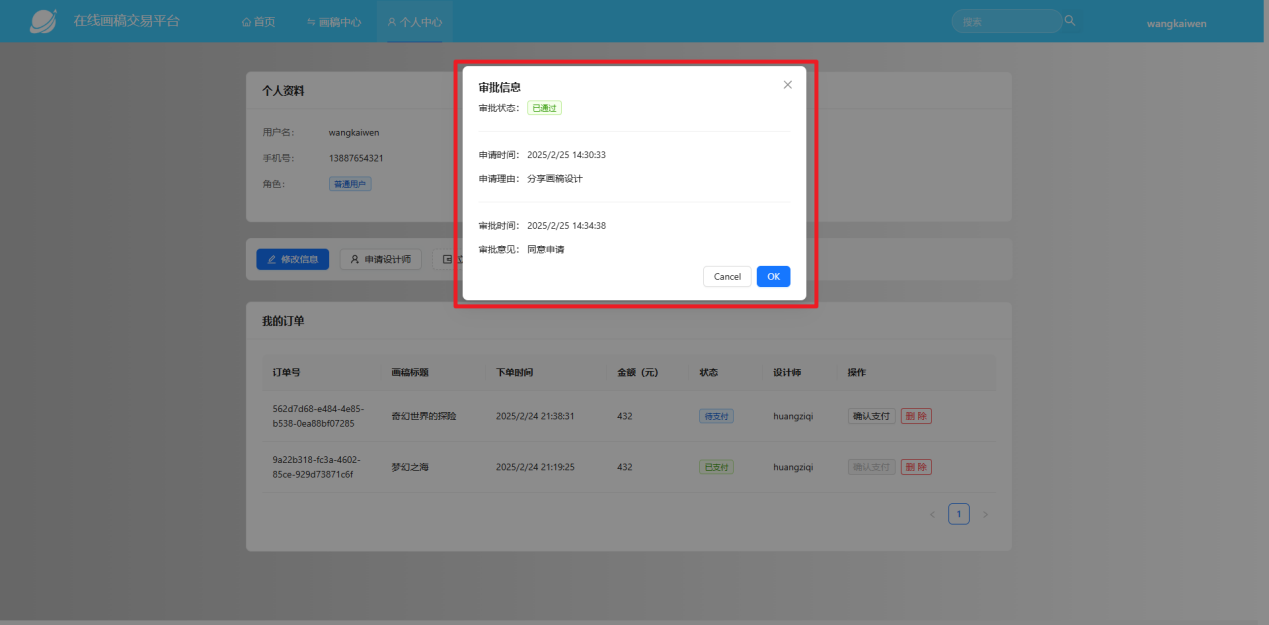


图5-9 用户查看审批结果

5.5 画稿交易

在本系统中，设计师通过发布画稿、设置价格、提供描述等信息来进行画稿交易。普通用户可以浏览画稿、选择自己喜欢的内容，并完成购买。购买的过程中，系统会通过扣除买方余额并增加卖方余额来完成交易。在交易完成后，如果卖方希望提现其余额，则需要通过第三方支付平台（如支付宝）进行提现。

为了确保交易的顺利进行和用户体验，系统设计了详细的订单流程和交易管理机制，特别是在订单取消、退款等方面进行了严格的权限控制。接下来将详细介绍订单模型的设计、交易过程、取消订单的条件以及管理员退款的操作流程。流程如图5-10所示：

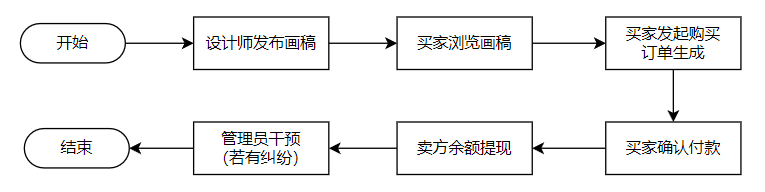


图5-10 画稿交易流程

5.5.1 交易流程

画稿交易流程由以下几个步骤组成：

（1）订单创建与支付：用户在浏览画稿后，决定购买某个画稿，系统会创建一个订单，并将订单状态设为 pending（待支付）。

（2）订单支付确认：买方支付后，系统会从买方账户扣除相应金额，并增加卖方账户余额。支付成功后，订单状态会更新为 completed（已完成）。

（3）订单取消：用户可以在订单未支付时取消订单。一旦订单支付完成，任何人（除了管理员）都无法取消订单。

（4）退款操作：退款操作只能由管理员执行，且只能对已支付的订单进行退款，退款操作会将金额退还给买方，并恢复卖方的余额。

5.5.2 订单创建与支付

在用户购买画稿时，首先创建一个订单记录。订单初始状态为 pending，表示待支付。在支付成功后，系统将自动扣除买方余额并增加卖方余额，订单状态更新为 completed。确认订单页面如图5-11所示，确认支付页面如图5-12所示：

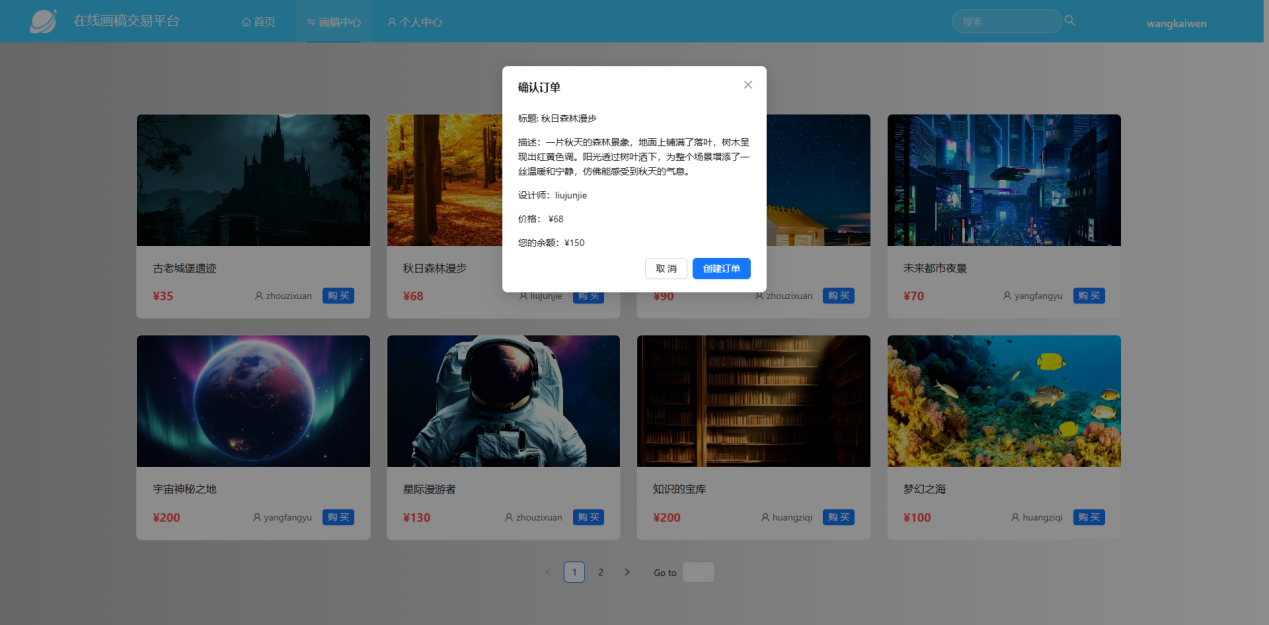


图5-11 确认订单页面

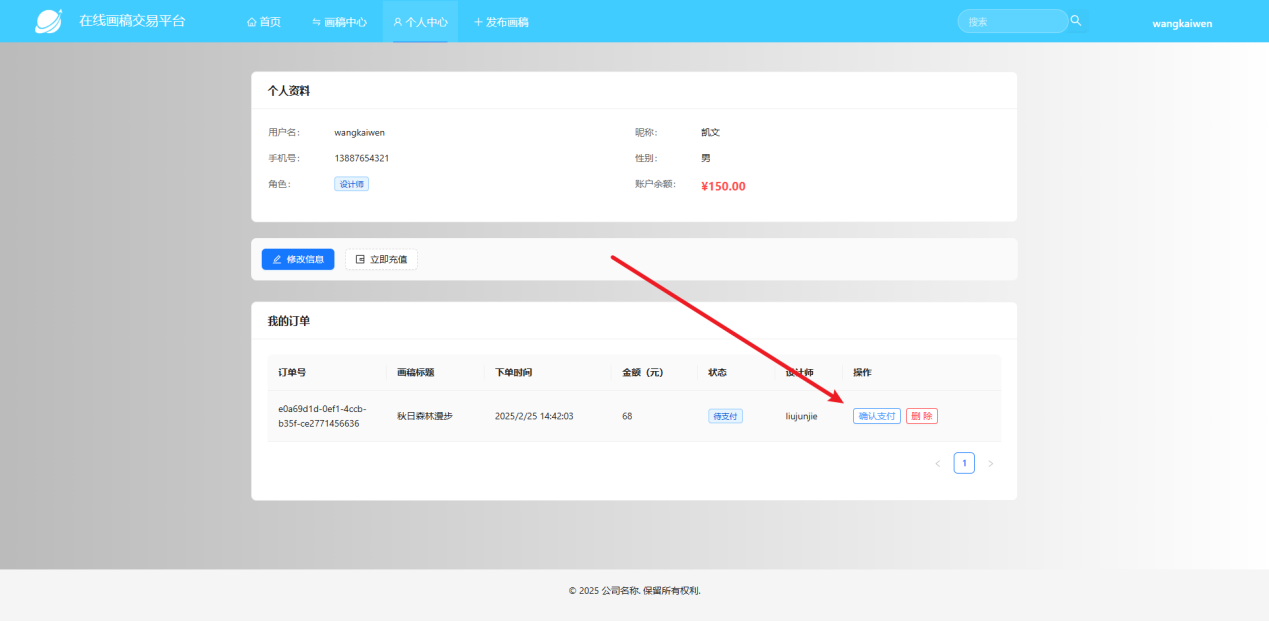


图5-12 确认支付页面

以下是实现该功能的代码：

def pay(self, order\_id):

order = Order.objects.get(id=order\_id)

# 查询画稿详情

draft\_id = order.draft\_id

draft = Draft.objects.get(id=draft\_id)

# 查询买方详情

user\_id = order.user\_id

user = User.objects.get(id=user\_id)

# 如果买方的余额小于画稿的价格，则交易失败

if user.balance < draft.price:

raise BusinessException("余额不足")

user.balance -= draft.price

user.save()

# 查询设计师详情，并将amount的金额转入设计师的余额中

designer\_id = draft.designer\_id

designer = User.objects.get(id=designer\_id)

designer.balance += draft.price

designer.save()

# 修改订单状态为“已支付”

order.status = OrderStatus.PAID.value

order.save()

5.5.4 管理员退款操作

管理员可以对已支付的订单进行退款操作，退款时系统会将交易金额退还给买方，并恢复卖方的余额。退款操作只能由管理员进行，且只能对 completed（已完成）状态的订单进行退款，页面如图5-13所示：

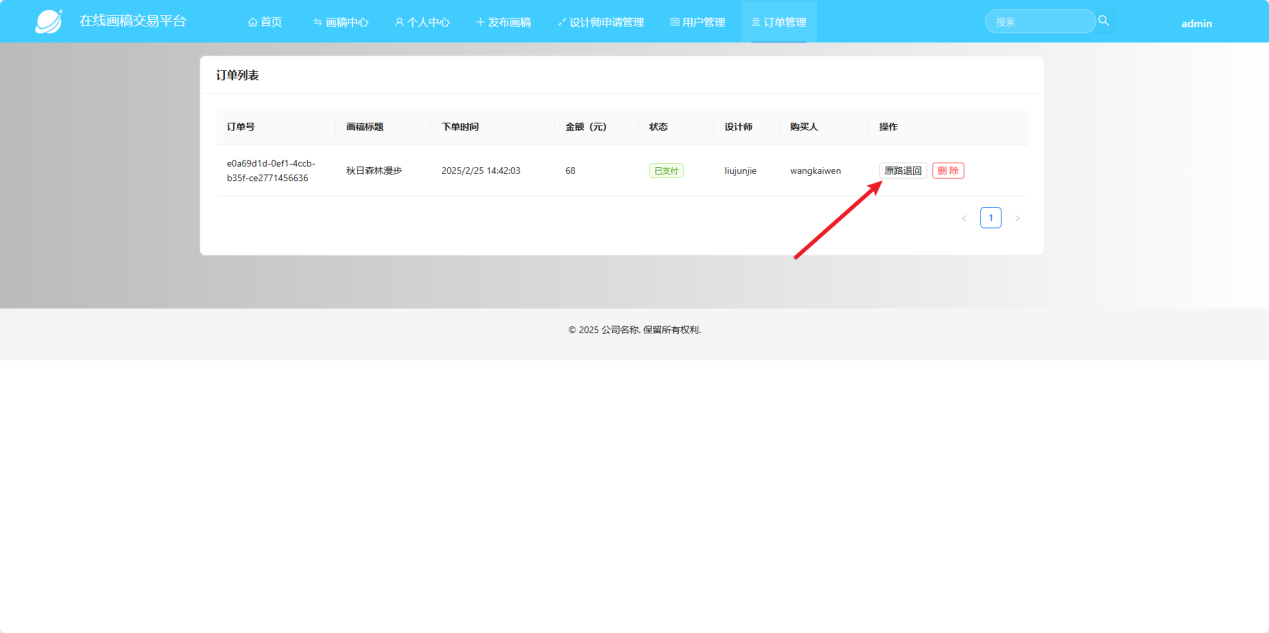


图5-13 退款页面

以下是管理员退款操作的代码：

def return\_order(self, order\_id):

order = Order.objects.get(id=order\_id)

# 查询画稿详情

draft\_id = order.draft\_id

draft = Draft.objects.get(id=draft\_id)

# 查询设计师详情

designer\_id = draft.designer\_id

designer = User.objects.get(id=designer\_id)

# 如果设计师的余额小于画稿的价格，则退回失败

if designer.balance < draft.price:

raise BusinessException("余额不足")

designer.balance -= draft.price

designer.save()

# 查询买方详情，，并将amount的金额买方的余额中

user\_id = order.user\_id

user = User.objects.get(id=user\_id)

user.balance += draft.price

user.save()

# 修改订单状态为“待支付”

order.status = OrderStatus.PENDING.value

order.save()

# 系统测试

6.1测试的意义及目标

系统测试是确保软件在开发完成后能够满足设计要求并在实际环境中稳定运行的关键步骤。在本项目中，测试不仅仅是一个验证过程，更是保证平台功能的可靠性、性能和安全性的必要手段。测试的目标是在不同的使用场景中发现潜在的问题，并对系统进行优化，以确保用户获得良好的体验，同时保证系统在处理高负载情况下的稳定性。

6.2测试方法

本系统采用多种测试方法，以确保系统的各项功能、性能和安全性都能够达到预期标准。测试方法的选择基于项目需求以及不同测试阶段的目标，目的是全面评估系统的稳定性、兼容性、性能等各方面的表现。以下是本项目中使用的主要测试方法：

6.2.1 功能测试

功能测试旨在验证系统的各项功能是否按照需求文档的要求正常工作，确保系统的每个模块都能实现其预定功能。功能测试的方法包括：

（1）单元测试：最基础的功能测试，主要验证每个独立模块的功能是否正确。通过编写测试用例，逐个验证系统中的小功能单元（如用户注册、登录、余额充值等）是否按照预期正常执行。

（2）集成测试：用于验证多个模块之间的交互是否正常。在功能测试阶段，我们将多个模块串联起来进行综合测试，确保整个业务流程（如从用户注册到交易完成的整个过程）能够顺利进行。

（3）接口测试：针对前后端分离的系统，接口测试尤为重要。通过验证前端和后端之间的请求和响应，确保数据正确传递和处理。使用 Postman 工具编写和执行API接口测试，确保用户请求能够被正确处理。

6.2.2 性能测试

性能测试用于评估系统在高负载情况下的表现，确保在高并发访问的情况下系统能够稳定运行。性能测试方法包括：

（1）负载测试：用于验证系统在高负载情况下的响应速度和稳定性。通过模拟多个用户同时访问系统，我们可以检查系统的响应时间、资源消耗等性能指标。常用工具包括 JMeter 和 Locust，这些工具能够帮助模拟并发用户，评估系统的承载能力。

（2）压力测试：用于在极端负载条件下测试系统的稳定性。例如，通过增加并发请求的数量，逐步压垮系统，看看系统在达到性能瓶颈时的表现。这有助于发现系统潜在的瓶颈和缺陷。

（3）容量测试：帮助验证系统在特定硬件和配置下，最大支持的用户数和事务数。通过模拟不同数量的并发用户，测试系统能够支持的最大并发数。

6.2.3 安全性测试

安全性测试旨在发现系统中的潜在安全漏洞，确保用户的敏感信息（如账号、密码、支付信息等）得到充分保护。常见的安全性测试方法包括：

（1）SQL注入攻击测试：测试系统是否存在SQL注入漏洞，即攻击者通过在输入框中插入恶意SQL语句来篡改数据库。我们通过模拟攻击输入，检查系统是否能够正确防范此类攻击。

（3）XSS攻击测试：测试系统是否存在跨站脚本攻击（XSS）漏洞，即攻击者通过在输入框中插入恶意JavaScript脚本来窃取用户信息或篡改页面内容。通过测试输入框、评论框等处的脚本注入，确保系统能够有效防止XSS攻击。

（4）CSRF攻击测试：测试系统是否存在跨站请求伪造漏洞。攻击者通过诱导用户点击恶意链接，发起未经授权的请求。我们通过模拟用户的恶意操作，验证系统是否具有有效的CSRF防护措施。

（5）认证和授权测试：测试系统的认证与授权机制，确保用户只有在通过身份验证后，才能访问特定的资源或操作。通过检查用户权限、管理权限等，确保系统在用户访问控制方面没有漏洞。

6.3功能性测试

测试内容覆盖了系统的所有核心功能模块，确保用户的每个操作和需求都能得到有效支持。功能性测试的目标是验证系统的功能是否符合预期，业务流程是否顺畅，用户操作是否符合实际需求。

6.3.1 测试目标

1. 确保用户注册、登录和认证功能正常。
2. 验证设计师认证及审核流程是否顺利执行。
3. 测试用户能够浏览画稿、进行购买、下单、支付等操作。
4. 确认用户余额管理、充值、提现等功能的正确性。
5. 保证订单管理流程，包括创建订单、支付、取消订单等功能能够顺利执行。
6. 验证管理员对用户、订单、画稿等的管理功能是否正确。

6.3.2 测试用例

以下是主要功能模块的功能性测试用例表，涵盖了各区块的关键功能和测试步骤。每个用例都包括了输入、期望输出以及测试步骤。功能性测试用例如表6-1所示：

表6-1 功能性测试用例表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **用例编号** | **功能模块** | **测试项** | **输入** | **预期输出** | **测试步骤** |
| TC001 | 用户注册 | 注册新用户 | 用户名、密码、邮箱 | 注册成功，返回200状态码 | 1. 填写注册信息 2. 提交表单   3. 检查返回值 |
| TC002 | 用户登录 | 用户登录 | 用户名、密码 | 登录成功，返回JWT Token | 1. 输入正确的用户名和密码   2. 登录成功并返回Token |
| TC003 | 设计师认证 | 普通用户申请成为设计师 | 用户ID、申请理由 | 提交申请，管理员审核中 | 1. 登录普通用户账号 2. 提交设计师申请   3. 返回申请提交成功 |
| TC004 | 设计师认证 | 管理员审核设计师申请 | 用户ID、审核状态（通过/拒绝） | 申请状态变更为“设计师” | 1. 登录管理员账号 2. 审核设计师申请   3. 确认状态变更 |
| TC005 | 余额充值 | 用户充值 | 充值金额、支付宝支付信息 | 充值成功，账户余额增加 | 1. 登录用户 2. 提交充值请求   3. 检查余额增加 |
| TC006 | 余额提现 | 用户申请提现 | 提现金额、支付宝账户信息 | 提现成功，提现金额减少 | 1. 登录设计师账号 2. 提交提现请求   3. 检查余额减少 |
| TC007 | 画稿发布 | 设计师发布画稿 | 画稿标题、描述、价格、图片URL | 画稿发布成功，返回画稿信息 | 1. 登录设计师账号 2. 填写画稿信息并提交   3. 返回画稿信息 |
| TC008 | 画稿购买 | 普通用户购买画稿 | 画稿ID、支付信息 | 支付成功，余额扣除，订单创建 | 1. 登录普通用户 2. 浏览画稿   3. 选择并支付画稿 |
| TC009 | 订单管理 | 买方取消未支付订单 | 订单ID | 订单取消，状态变更为“已取消” | 1. 登录买方账号 2. 选择订单并取消   3. 检查订单状态 |
| TC010 | 订单管理 | 卖方查看订单 | 订单ID | 订单信息正确显示 | 1. 登录卖方账号 2. 查看订单信息   3. 确认订单信息准确 |
| TC011 | 订单管理 | 管理员退款操作 | 订单ID、退款金额 | 退款成功，订单状态变更为“已退款” | 1. 登录管理员账号 2. 查询订单并进行退款   3. 确认订单退款状态 |
| TC012 | 系统安全性 | 防止SQL注入 | 恶意SQL注入 | 应用拒绝恶意输入 | 1. 输入SQL注入代码到用户注册或登录表单   2. 确认系统拒绝此类输入 |
| TC013 | 系统安全性 | 防止XSS攻击 | 恶意JavaScript脚本 | 系统无响应，XSS攻击无法执行 | 1. 输入XSS攻击脚本到评论框或搜索框   2. 确认系统拒绝此类脚本 |

6.3.3 功能测试详细描述

（1）用户注册与登录

功能测试开始时，首先会验证系统的注册与登录功能。用户能够通过提供有效的用户名、密码以及邮箱信息进行注册，并且可以通过用户名和密码成功登录获取JWT Token，后续操作将使用该Token进行身份验证。

（2）设计师认证与审核流程

测试用户是否能够提交设计师认证申请，以及管理员是否能够审核和批准该申请。普通用户提交申请后，管理员通过后台审核功能，批准通过后，用户将成为设计师。

（3）余额充值与提现功能

充值功能的测试检查用户是否能够成功充值余额，并且系统是否正确处理充值请求。提现功能测试则验证用户是否能够顺利申请提现，提现请求将通过支付宝进行处理。

（4）画稿发布与购买流程

设计师发布画稿后，普通用户能够浏览、选择并购买画稿，测试确认购买流程是否完整。交易过程中，买方账户余额会减少，卖方账户余额增加。

（5）订单管理功能

测试订单的创建、查看、取消和退款功能。只有管理员能够进行退款操作，买方在确认支付后无法取消订单。未支付订单在一定条件下允许取消，退款操作仅限管理员执行。

（6）安全性与异常处理

通过模拟SQL注入和XSS攻击，确保系统具备防范常见安全漏洞的能力。系统应当能够正确过滤恶意输入，防止任何可能的攻击。

6.4测试结论

通过本次系统的功能性测试，系统整体功能满足设计要求，各模块的关键功能正常运作。测试过程中，我们重点验证了用户注册、登录、设计师认证、余额充值、余额提现、画稿发布、订单管理等核心业务流程。测试用例覆盖了正常流程以及常见的异常情况，确保系统的稳定性和安全性。

**主要测试结论：**

（1）用户注册与登录：用户注册、登录功能均能正常工作，用户能够顺利注册账户并通过用户名与密码登录。JWT令牌的生成和验证功能也表现良好。

（2）设计师认证流程：普通用户提交设计师认证申请后，管理员可进行审核并决定是否通过或拒绝申请。审批流程顺利，信息反馈及时，设计师权限的授予也得到了有效验证。

（3）余额管理与充值提现：用户能够顺利进行余额充值与提现操作，充值和提现功能准确执行，支付接口（如支付宝）集成良好，确保用户账户余额的实时更新。

（4）画稿交易流程：设计师能够发布画稿并设置价格，普通用户可以浏览、选择并购买画稿。购买后，交易成功，订单创建并扣除相应的金额。买方可在支付前取消订单，而支付后订单状态为不可取消，确保了交易的安全性。

（5）订单管理：订单的创建、状态更新以及订单取消功能正常。管理员具有退款权限，可以为买方或卖方处理退款请求，确保系统的灵活性和公平性。

（6）系统安全性：在输入有效性方面，系统防止了SQL注入和XSS攻击等常见的安全漏洞，保护了用户数据和系统的安全性。

总的来说，本次系统测试达到了预期目标，系统整体运行稳定，功能实现符合需求，具备了上线的条件。

**结语**

本文设计并实现了一个基于Web的在线画稿交易平台，围绕着设计师认证、画稿交易、用户管理、支付系统等核心功能展开，结合了现代技术如Django框架、JWT认证、支付宝支付等，实现了一个完整的、功能丰富的交易系统。通过采用前后端分离架构，使得系统具有较高的可扩展性和灵活性；通过引入设计师认证机制和完善的交易管理系统，提升了平台的安全性与用户体验。

在开发过程中，系统采用了多种安全性保障手段，确保用户数据的隐私性与交易的安全性。例如，利用JWT对用户身份进行认证与授权，保证了每个请求的安全性；同时在支付环节集成了支付宝支付接口，简化了充值和提现操作，提升了平台的支付便利性。

未来，平台可以进一步扩展其功能，包括但不限于支持更多支付方式、完善用户评价体系、引入智能合约保障交易的公正性等。此外，随着用户规模的扩大，平台还可以引入更为复杂的算法，如推荐系统，以提高用户的交易体验和平台的运营效率。

**致谢**

求学的时光如流沙悄然而逝，行笔至此，我的脑海中涌现出无数记忆的片段。在此，我谨向所有支持、帮助和陪伴我的人致以最诚挚的感谢。

首先，我要衷心感谢我的导师XXX老师。从论文选题、技术选型到系统实现的每个阶段，X老师始终以严谨的学术态度和丰富的实践经验为我指明方向，总是能够及时且细心地给我解答疑惑并提供实质性的帮助。良师难得，我将一生以您为榜样。

其次，感谢我的父母和家人。在论文攻坚阶段，你们始终给予我无条件的理解与支持。每当我因技术难题而焦虑时，是你们的鼓励让我重拾信心，你们是我求学路上最坚实的后盾。

最后，感谢我的同事，在艺术领域给我提供了宝贵的专业支持。

回首这段旅程，我深知论文的完成绝非仅靠一己之力。正是师长、同事与家人的支持，让我在探索技术的道路上不断突破自我。未来，我将带着这份感恩之心，继续在计算机领域深耕前行。谨以这篇青涩的论文，致敬所有在数字荒原上播种星空的人。

**参考文献**

[1]袁小笛.市场的利与弊—围绕水彩画艺术电商平台的探讨和分析[D].湖南师范大学,2016.

[2]焦阳.我国数字版权交易平台建设研究[D].河北大学,2017.

[3]杜娟.学生艺术作品展示平台的设计与实现[D].大连理工大学,2016.

[4]李刚.Python语言下校园物品交易系统[J].马鞍山师范高等专科学校<滨江校区>.DOI:10.16184/j.cnki.comprg.2024.11.026

[5]卢万有.基于JWT的RBAC在前后端分离项目中的设计与实现[J].福建省长汀职业中专学校.DOI:10.16184/j.cnki.comprg.2025.01.004

[6]陈炯.陈周云.潘锦锦.吴光华.王国锋.基于Django和Vue的试验策划管理系统设计与实现[J].吉利汽车研究院(宁波)有限公司.DOI:10.19850/j.cnki.2096-4706.2024.19.005

[7]潘涛.王柳.董冉冉.基于Vue.js框架的网上商城管理系统的设计与实现[J].安徽工程大学.DOI:10.15913/j.cnki.kjycx.2023.13.003

[8]李爽.高校闲置物品交易系统的设计与实现[J].贵州财经大学

[9]张宇.面向校园的闲置资源交易系统[D].苏州大学

[10]Park S , Park S A ,Yun, Sang-PilLee, KyungeunKang, ByeongkwanChoi, Myeong-inJang, HyeonwooPark, Sehyun.Design and Implementation of a Futuristic EV Energy Trading System (FEETS) Connected with Buildings, PV, and ESS for a Carbon-Neutral Society[J].buildings, 2023, 13(3).

[11]刘叶.农业公司果蔬批发信息管理系统的设计与实现[D].首都经济贸易大学,2021.

[12]S Saxena，H Farag，A Brookson，H Turesson，HM Kim.Design and Field Implementation of Blockchain Based Renewable Energy Trading in Residential Communities[J]. 2019.DOI:10.48550/arXiv.1907.12370.