格式要求：

1. 正文中文用宋体，英文与数字用Times New Roman，均为小四号。正文行间距固定值20磅。
2. 一级标题段前段后间距1行，二级标题段前段后间距0.5行
3. **系统体系结构设计**

2.1 系统特点分析

本系统为典型的基于事物交互的Web平台（B/S架构），聚焦学生高频、低成本、高信任度的二手书交易需求，通过用户、商品、交易的多维数据联动，实现二手书资源的优化配置。核心特点如下：

1. 精准校园场景适配：

​ 课程教材智能匹配：对接学校选课系统API，自动推荐相关课程教材（如《高等数学-李老师2023版》）。

​ 支持校园内二手书商：与校园内书商合作，提高课本回收速度。

2. 极简交易流程：

​ 扫码卖书：通过ISBN扫码自动填充书籍信息（含课程/版本备注），30秒完成上架。

搜索匹配：通过课程代码，匹配课程教材

1. 提升管理透明度

对书籍状态（新旧程度、价格波动、交易历史）、用户信用评级等数据进行统一管理，规范平台交易秩序。

2.2 系统体系结构设计

2.2.1 系统体系结构模式

本系统采用三层架构分层设计与分布式架构，基于B/S（Browser/Server）模式实现，将系统分为浏览器端以及服务器端，系统由以下三个核心层级构成：

| **​层级** | **​主要功能** | **​运行方式** |
| --- | --- | --- |
| ​**前端** | 用户交互和数据显示 | ​基于浏览器运行 |
| **​后端** | 业务逻辑处理和数据存储 | 基于服务器部署运行 |
| **​数据库** | 数据的存储以及管理 | ​基于数据存储运行 |

表 2-1

2.2.2 系统体系结构设计

本系统采用三层架构设计模式，将不同功能模块分配到不同的层次上，保证系统的可扩展性和可维护性。通过将业务逻辑、数据管理和用户界面分离，使得修改其中任意部分不会对其他部分造成影响。

同时，系统的体系结构严格遵循软件设计基本原则，包括：高内聚低耦合：各模块功能独立，减少相互依赖，提高代码复用性。单一职责原则：每个模块仅负责一项核心功能，确保逻辑清晰、易于维护。这种分层设计使系统架构更加合理、清晰，便于后续功能扩展和代码维护。

（1）逻辑视图设计

本系统采用类图作为逻辑视图的核心表达方式，通过面向对象的方式呈现系统的主要实体及其交互关系。清晰定义了系统的核心业务实体及其交互逻辑，覆盖了从书籍发布、交易下单、物流跟踪到用户通信的全流程，同时通过状态枚举和权限控制保障系统的规范性与安全性。类图共包含七个核心类，分别对应系统的主要业务模块，各类间通过属性关联形成完整的业务逻辑链。

以下是我们所设计的核心类及关键属性/方法表格及类图：​

| 类名 | 关键属性/方法 | 核心关联关系 |
| --- | --- | --- |
| ​User​ | userID, username, status, updateProfile(), login() | 关联Order、Message、Notification |
| ​Book​ | bookID, title, status, publish() | 关联Order（卖家） |
| ​Order​ | orderID, status, paymentMethod, cancel(), requestRefund() | 关联User、Book、Shipping |
| ​Shipping​ | shippingID, address, status, updateStatus() | 关联Order（物流跟踪） |
| ​Message​ | messageID, content, send() | 关联User（收发双方） |
| ​Notification​ | notificationID, type, send(), isUrgent() | 关联User（接收方） |
| ​Admin​ | adminID, rolePermission | 关联User（管理权限） |

表 2-2

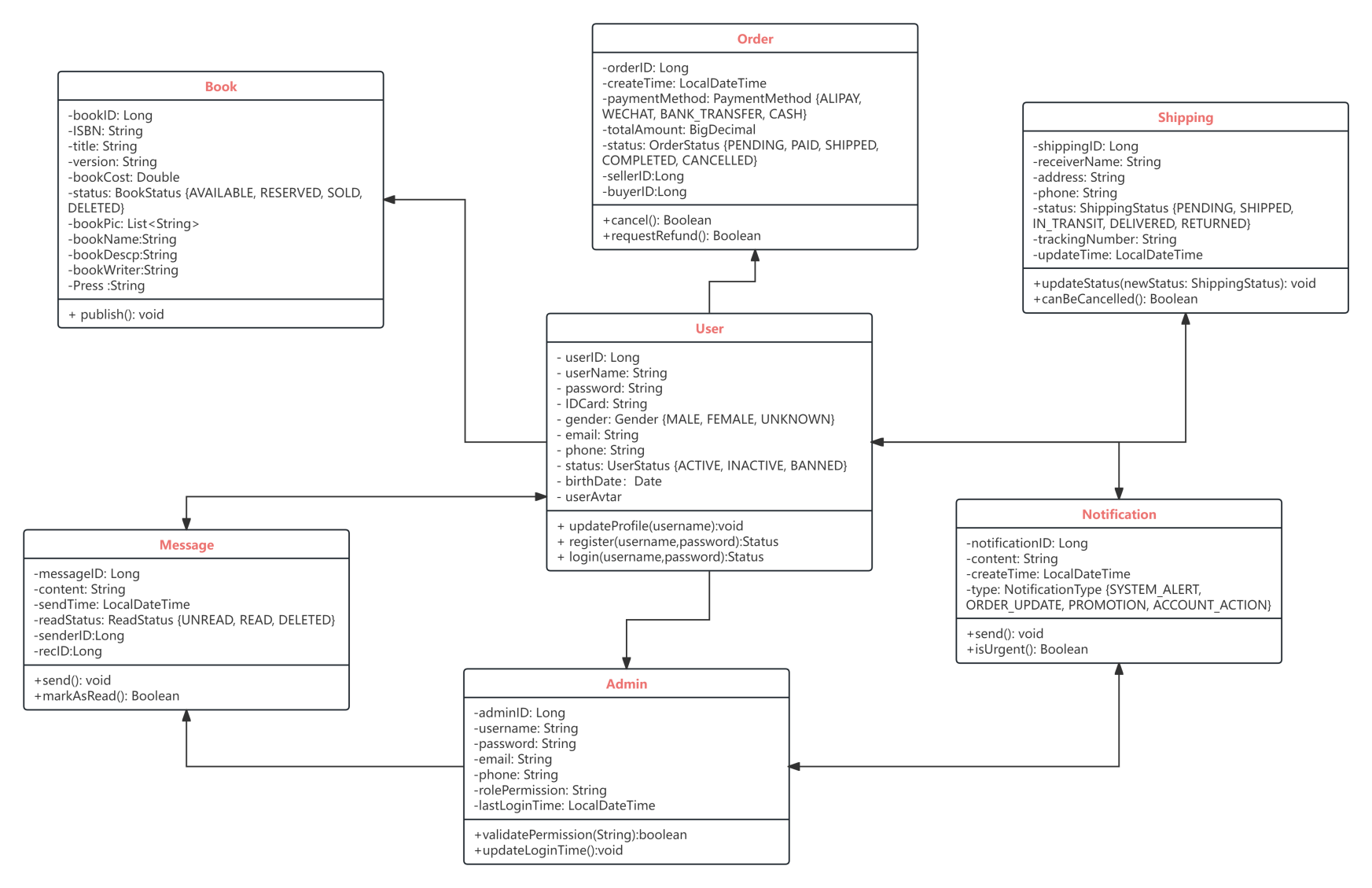


图 2-1

1. 开发及运行视图设计

我们的二手书交易系统采用了三层架构，并且结合MVC架构，将系统自上而下划分为VIEW层、Controller层、Service层、Dao层和数据层。各层职责明确，通过标准化接口实现解耦，确保系统的可维护性和可扩展性。为了便于说明，我们这里通过表格以及系统架构图的形式进行展示

| 层级 | 核心功能 | 对应模块（二手书交易系统实现） |
| --- | --- | --- |
| ​VIEW层​ | 用户交互界面，接收输入并展示结果 | - 登录界面 - 首页推荐（热门书籍展示） - 消息通知（交易提醒） - 书籍搜索（关键词/分类筛选） - 个人管理（用户资料修改） - 书籍贩卖（发布二手书） - 订单查询（交易记录） - 客服售后（纠纷处理） |
| ​Controller层​ | 请求路由与参数校验，调用Service层处理业务逻辑 | 接收前端请求，校验参数后转发至对应Service模块 |
| ​Service层​ | 核心业务逻辑实现，组合Dao层操作完成复杂功能 | - 用户认证服务（登录/注册） - 书籍推荐与搜索服务（个性化推荐） - 数据获取服务（书籍详情） - 消息服务（买卖双方通信） - 交易服务（订单创建与支付） - 数据统计服务（交易量分析） |
| ​Dao层​ | 数据持久化操作，封装数据库访问细节 | 提供对MongoDB（非结构化数据）、MySQL（结构化数据）、Redis（缓存） |
| ​数据层​ | 数据存储与缓存，支持高并发访问 | MongoDB：存储书籍图片、用户聊天记录等非结构化数据 MySQL：存储用户信息、订单数据等核心业务数据 Redis：缓存热门书籍、会话状态等高频访问数据 |

表 2-3

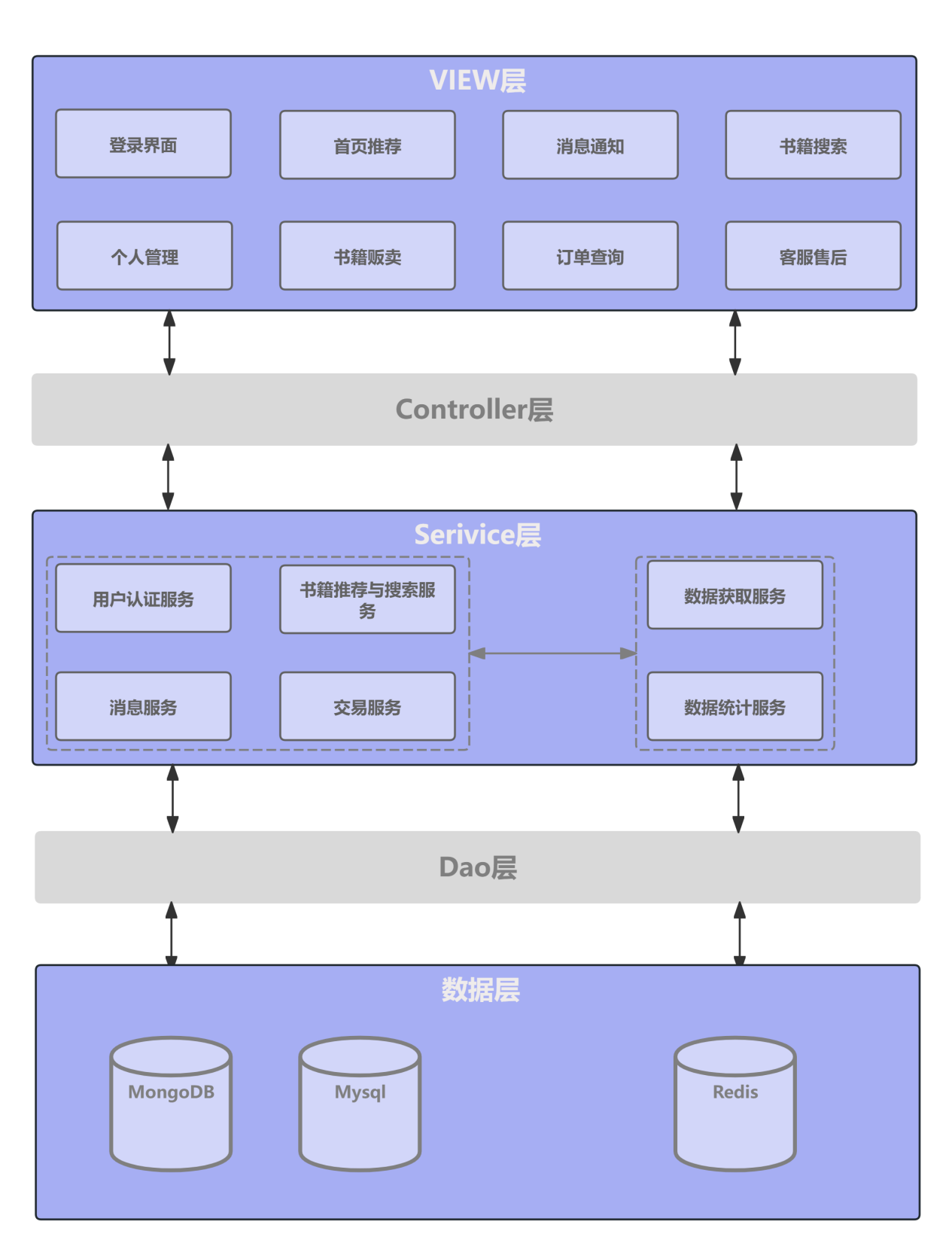


图 2-2

（3）部署视图设计

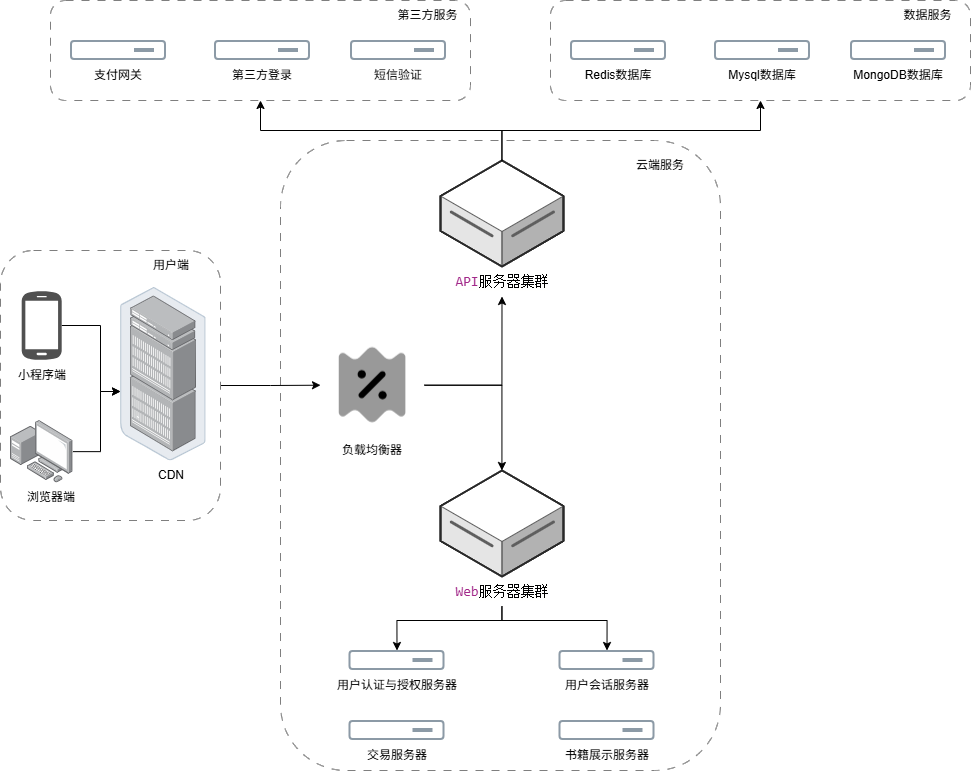
系统部署主要分为用户端以及云端两个部分。用户可以通过手机小程序端口或者电脑浏览器端口进行访问，用户请求首先会经过CDN加速，然后通过负载均衡器合理分配流量至云端。云端服务器主要分为两个部分：Web服务器集群以及API服务器集群。Web服务器集群主要负责网页内容展示以及拥有专门的功能服务器如：用户认证与授权服务器：管登录注册、用户会话服务器：记录用户登录状态等。而API服务器集群则主要是处理第三方服务支持以及数据库存储部分。

图 2-3