ssm基于Android校园二手交易APP软件需求说明书

# 1. 引言

## 1.1 文件信息

文件状态：已完成

文件标识：“基于SSM框架的校园二手交易APP需求说明书”

当前版本：1.0

作者：[具体姓名]

完成日期：2024年10月7日

版本更新信息：无

团队成员信息：

姓名：[团队成员1姓名]

学号：[团队成员1学号]

团队：校园二手交易APP开发团队

姓名：[团队成员2姓名]

学号：[团队成员2学号]

团队：校园二手交易APP开发团队

## 1.2 使用人员

学生：作为主要的用户群体，学生将使用该APP进行二手商品的买卖、浏览和交易管理。

商家：提供二手商品的卖家，他们需要利用该APP快速上架商品，并有效管理商品、订单和配送流程。

管理员：管理平台用户和商家，负责监督整个交易流程，处理用户投诉，管理用户账户和权限。

## 1.3 编写目的

本软件需求说明书旨在为基于SSM框架（Spring + SpringMVC + MyBatis）开发的校园二手交易APP提供详尽的功能需求和非功能需求说明。其主要目标是确保开发人员、测试人员、项目经理及其他相关人员对系统的功能需求有统一、清晰的理解，并为后续的开发、测试、部署及维护工作提供准确、可靠的依据。

## 1.4 背景

随着校园内二手商品交易需求的日益增长，现有的交易方式往往分散、不便，缺乏一个集中化、智能化的交易平台。本项目旨在开发一个基于SSM框架的Android校园二手交易APP，以满足学生和校园商家在二手商品交易中的实际需求。该系统将提供一个便捷、高效、安全的交易平台，涵盖商品搜索、购物车管理、下单支付、订单和配送管理等核心功能，旨在提升校园二手商品交易的效率和用户体验。

## 1.5 定义与缩写

SSM：Spring + SpringMVC + MyBatis框架，用于构建后端服务。

APP：Android应用程序，指校园二手交易APP。

API：应用程序接口，用于前后端数据交互。

MySQL：关系型数据库管理系统，用于存储用户、商品、订单等数据。

Redis：高性能缓存数据库，用于提高系统响应速度和数据处理能力。

Nginx：高性能的HTTP和反向代理web服务器，用于处理静态资源请求和负载均衡。

Tomcat：Apache软件基金会（ASF）开发的Servlet容器，用于部署Java Web应用。

## 1.6 参考资料

《Spring官方文档》：详细介绍Spring框架的使用方法和最佳实践。

《MyBatis用户指南》：全面解析MyBatis框架的功能和用法。

《Android开发文档》：Android官方提供的开发指南和API参考。

# 2. 任务概述

## 2.1 目标

本项目旨在开发一个基于SSM框架的Android校园二手交易APP，实现以下主要功能：

用户注册/登录功能

商品搜索功能

购物车管理功能

下单与支付功能

订单和配送管理功能

商家管理功能

用户管理功能

## 2.2 用户特点

学生用户：

熟悉手机操作，对界面友好性和易用性有较高要求。

期待一个易于使用的界面和便捷的交易流程。

关注商品的价格、质量、卖家信誉等信息。

商家用户：

希望快速上架商品，并有效管理商品信息、订单状态和配送流程。

期待系统简洁高效，能够支持批量操作和多商品管理。

关注系统的稳定性和数据安全性。

管理员用户：

需要一个后台系统来管理用户、商品和订单信息。

负责监督整个交易流程，处理用户投诉和交易异常。

关注系统的权限管理和数据安全控制。

## 2.3 应用环境

前端设备：本APP设计用于Android设备，兼容并支持Android 5.0及以上版本的各种智能手机和平板设备。这意味着，只要用户的设备运行的是Android 5.0或更高版本，他们就可以下载并安装本APP，享受二手商品交易的便捷服务。

服务器端：服务器端架构采用先进的SSM（Spring + SpringMVC + MyBatis）框架，这是一种流行的Java Web应用框架组合，提供了高度的灵活性和可扩展性。数据库系统选择的是MySQL，它是一款功能强大、易于使用的开源关系型数据库管理系统，能够满足大多数中小型应用的数据存储需求。此外，为了提高系统性能和响应速度，我们还引入了Redis作为缓存系统，它能够有效地存储和读取数据，减少数据库访问次数，从而提升整体性能。

网络要求：为了确保用户能够流畅地使用APP的各项功能，需要有一个稳定的网络连接。无论是Wi-Fi还是4G/5G网络，只要能够提供足够的带宽和低延迟，都可以满足APP的运行需求。在网络环境不稳定或中断的情况下，APP将尝试使用本地缓存的数据来保持部分功能的可用性，同时提示用户检查网络连接。

## 2.4 假定和约束

在系统设计和开发过程中，我们基于一些合理的假定和约束来制定决策。首先，我们假定系统必须能够处理高并发下的订单交易，这是因为在校园内，特别是在一些特殊的时期（如学期末、毕业季等），二手商品的交易需求可能会激增，系统需要能够承受这种压力。其次，我们假定所有支付交易都通过安全的第三方支付接口完成，以确保用户资金的安全性和合规性。此外，我们还假定数据必须进行加密存储，以保护用户隐私和敏感信息不被泄露。

在约束方面，我们主要面临的是技术选型、资源限制和法律法规等方面的限制。技术选型方面，我们已经确定了使用SSM框架、MySQL数据库和Redis缓存等关键技术，这在一定程度上限制了我们在技术上的选择。资源限制方面，我们需要在有限的预算和时间内完成系统的开发、测试和部署工作，因此需要在功能需求、性能需求和开发成本之间做出权衡。法律法规方面，我们需要遵守相关的法律法规要求，如用户隐私保护、数据安全和电子支付等方面的规定。

# 3. 需求规定

## 3.1 对功能的规定

3.1.1 用户注册/登录功能

用户注册/登录功能是APP的基础功能之一。用户可以通过手机号或邮箱进行注册，注册过程中需要填写一些基本信息（如用户名、密码、验证码等），并进行手机或邮箱验证以确保账户的真实性。注册成功后，用户可以使用注册时填写的手机号或邮箱及密码进行登录。此外，为了提升用户体验和安全性，我们还支持短信验证码登录方式。用户只需输入手机号并获取验证码，即可快速登录APP。

3.1.2 搜索功能

搜索功能是APP的重要功能之一。用户可以通过商品分类和关键字进行搜索，快速找到他们感兴趣的二手商品。搜索结果将展示商品的基本信息（如商品名称、价格、图片、卖家信息等），以便用户能够快速浏览和筛选。为了提高搜索的准确性和效率，我们还支持模糊搜索和智能推荐功能。模糊搜索允许用户输入不完整的关键字进行搜索，系统会根据相似度给出相关的搜索结果。智能推荐功能则根据用户的浏览历史和购买记录，推荐可能感兴趣的商品给用户。

3.1.3 购物车功能

购物车功能是APP的核心功能之一。用户可以将他们感兴趣的商品添加到购物车中，以便在后续进行批量购买或管理。购物车支持批量删除和修改数量等操作，用户可以随时查看购物车中的商品列表、总价和优惠信息等。此外，购物车还支持与下单功能的无缝对接，用户可以直接从购物车进入下单页面进行支付操作。

3.1.4 下单功能

下单功能是APP的关键功能之一。用户可以从购物车中选择要购买的商品，填写收货信息（如收货地址、联系电话等），并选择支付方式（如支付宝、微信支付等）进行支付。在下单过程中，系统会对订单信息进行校验和审核，以确保订单的有效性和合规性。一旦订单被确认有效并支付成功，系统将生成一个唯一的订单号，并通知卖家进行发货操作。同时，用户还可以在订单管理页面中查看订单的详细信息、物流信息和支付状态等。

3.1.5 支付功能

支付功能是APP的重要安全保障之一。我们集成了多种常用的支付方式（如支付宝、微信支付等），用户可以根据自己的喜好和习惯选择适合自己的支付方式。在支付过程中，系统会对支付信息进行加密处理，并与第三方支付接口进行交互以完成支付操作。支付成功后，系统将更新订单状态并通知卖家进行后续操作。同时，用户还可以在支付结果页面中查看支付详情和支付凭证等信息。

3.1.6 订单管理功能

订单管理功能是APP的重要功能之一。用户可以在订单管理页面中查看自己的历史订单信息（如订单号、商品名称、价格、购买时间、物流信息等）。对于尚未完成的订单。

# 4. 运行环境规定

## 4.1 设备

客户端：APP支持在Android 5.0及以上版本的设备上运行。这确保了大多数现代Android设备都能安装和使用该APP。

服务器端：服务器端部署在Linux服务器上，优先选择云平台（如阿里云、AWS等），以确保系统的高可用性和可扩展性。

## 4.2 支持软件

服务器软件：使用Nginx或Tomcat作为服务器容器，以支持Java应用的部署和运行。

数据库：采用MySQL数据库存储商品、用户、订单等数据，确保数据的可靠性和持久性。

缓存：使用Redis进行缓存优化，提高系统响应速度和用户体验。

支付接口：集成支付宝、微信支付等第三方支付接口，支持用户进行安全便捷的支付操作。

## 4.3 接口

API接口：APP通过RESTful API与服务器端进行交互，实现数据的请求和响应。这有助于保持前后端的分离和独立开发，提高系统的灵活性和可维护性。

第三方接口：除了支付接口外，APP还可能需要与其他第三方服务（如物流服务、短信服务等）进行集成，以满足用户的多样化需求。

## 4.4 控制

权限管理：通过权限管理功能，确保管理员、商家和用户分别拥有不同的操作权限。这有助于防止未授权访问和操作，保护系统的安全性和稳定性。

日志记录：对系统的重要操作进行日志记录，以便在出现问题时进行排查和追踪。同时，日志记录也有助于审计和合规性检查。