ChatGPT

小米商城项目说明文档

项目概述与设计基础

小米商城项目基于开源简易商城系统 EmarketMall（JSP/Servlet 架构）进行改造，实现商品浏览购买、用户账户和售后服务等功能 [1] [2]。本项目将 EmarketMall 的 JSP 项目基础结构作为实现基础，并在其上新增“设备服务”（售后服务）模块 [3] [4]。为了简化架构，**所有数据库的增删查改操作集中在一个 `Model.java` 类中**，不再采用多层 DAO 分离的设计，以方便后续通过 AI 或代码生成器进行操作。

数据库结构及建表 SQL

小米商城数据库在 EmarketMall 原有数据库结构基础上扩展，采用 MySQL 8.x。 [5] [6] 原有表涵盖用户、商品、订单等，新增 **设备信息表** 和 **售后申请表** 以支持设备绑定和售后服务功能 [6] [7]。所有表以自增整数型主键 `id` 标识，并包含通用维护字段（创建时间、修改时间、删除标记等） [8]。下表列出了主要的数据表结构及字段：

```
-- 1. 用户信息表：存储前台注册用户
DROP TABLE IF EXISTS `user_info`;
CREATE TABLE `user_info` (
 `delFlag` int DEFAULT NULL COMMENT '删除标记',
 `createdBy` varchar(32) DEFAULT NULL COMMENT '创建人',
 `createdTime` datetime DEFAULT NULL COMMENT '创建时间',
 `updatedBy` varchar(32) DEFAULT NULL COMMENT '更新人',
 `updatedTime` datetime DEFAULT NULL COMMENT '更新时间',
 `remark` varchar(255) DEFAULT NULL COMMENT '备注',
 `id` int NOT NULL AUTO_INCREMENT COMMENT '用户ID',
 `name` varchar(255) NOT NULL COMMENT '用户姓名',
 `nickname` varchar(255) DEFAULT NULL COMMENT '用户昵称',
 `loginName` varchar(255) NOT NULL COMMENT '登录名',
 `avatar` varchar(900) DEFAULT NULL COMMENT '头像URL',
 `email` varchar(255) DEFAULT NULL COMMENT '邮箱',
 `phone` varchar(255) NOT NULL COMMENT '联系电话',
 `password` varchar(255) NOT NULL COMMENT '密码(MD5加密)',
 PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COMMENT='用户信息表';

-- 2. 收货地址表：存储用户的收货地址
DROP TABLE IF EXISTS `delivery_address`;
CREATE TABLE `delivery_address` (
 `id` int NOT NULL AUTO_INCREMENT COMMENT '地址ID',
 `user_id` int NOT NULL COMMENT '用户ID',
 `receiver_name` varchar(255) NOT NULL COMMENT '收货人姓名',
 `phone` varchar(20) NOT NULL COMMENT '联系电话',
 `province` varchar(100) NOT NULL COMMENT '省份',
 `city` varchar(100) NOT NULL COMMENT '城市',
 `district` varchar(100) NOT NULL COMMENT '区/县',
```

```
  `detail_address` varchar(255) NOT NULL COMMENT '详细地址',
  `is_default` tinyint(1) DEFAULT 0 COMMENT '是否默认地址',
  `delFlag` int DEFAULT NULL COMMENT '删除标记',
  `createdBy` varchar(32) DEFAULT NULL COMMENT '创建人',
  `createdTime` datetime DEFAULT NULL COMMENT '创建时间',
  `updatedBy` varchar(32) DEFAULT NULL COMMENT '更新人',
  `updatedTime` datetime DEFAULT NULL COMMENT '更新时间',
  `remark` varchar(255) DEFAULT NULL COMMENT '备注',
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COMMENT='收货地址表';

-- 3. 部门表：存储后台管理员所属部门信息
DROP TABLE IF EXISTS `department`;
CREATE TABLE `department` (
  `delFlag` int DEFAULT NULL COMMENT '删除标记',
  `createdBy` varchar(32) DEFAULT NULL COMMENT '创建人',
  `createdTime` datetime DEFAULT NULL COMMENT '创建时间',
  `updatedBy` varchar(32) DEFAULT NULL COMMENT '更新人',
  `updatedTime` datetime DEFAULT NULL COMMENT '更新时间',
  `remark` varchar(255) DEFAULT NULL COMMENT '备注',
  `id` int NOT NULL AUTO_INCREMENT COMMENT '部门ID',
  `name` varchar(255) NOT NULL COMMENT '部门名称',
  `phone_num` varchar(255) DEFAULT NULL COMMENT '联系电话',
  `leader` varchar(255) NOT NULL COMMENT '部门负责人',
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COMMENT='部门表';

-- 4. 员工信息表：存储后台管理员账号
DROP TABLE IF EXISTS `staff_info`;
CREATE TABLE `staff_info` (
  `delFlag` int DEFAULT NULL COMMENT '删除标记',
  `createdBy` varchar(32) DEFAULT NULL COMMENT '创建人',
  `createdTime` datetime DEFAULT NULL COMMENT '创建时间',
  `updatedBy` varchar(32) DEFAULT NULL COMMENT '更新人',
  `updatedTime` datetime DEFAULT NULL COMMENT '更新时间',
  `remark` varchar(255) DEFAULT NULL COMMENT '备注',
  `id` int NOT NULL AUTO_INCREMENT COMMENT '员工ID',
  `login_name` varchar(255) NOT NULL COMMENT '登录名',
  `name` varchar(255) NOT NULL COMMENT '员工姓名',
  `staff_type` varchar(255) NOT NULL COMMENT '员工类型/岗位',
  `email` varchar(255) DEFAULT NULL COMMENT '邮箱',
  `phone_num` varchar(255) NOT NULL COMMENT '电话',
  `gender` varchar(255) DEFAULT NULL COMMENT '性别',
  `avatar` varchar(255) DEFAULT NULL COMMENT '头像URL',
  `password` varchar(255) NOT NULL COMMENT '密码',
  `status` int NOT NULL COMMENT '帐号状态(0正常,1停用)',
  `dept_id` int NOT NULL COMMENT '所属部门ID',
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COMMENT='后台员工表';

-- 5. 商品分类表：支持多级商品分类
```

```sql
DROP TABLE IF EXISTS `product_category`;
CREATE TABLE `product_category` (
  `delFlag` int DEFAULT NULL COMMENT '删除标记',
  `createdBy` varchar(32) DEFAULT NULL COMMENT '创建人',
  `createdTime` datetime DEFAULT NULL COMMENT '创建时间',
  `updatedBy` varchar(32) DEFAULT NULL COMMENT '更新人',
  `updatedTime` datetime DEFAULT NULL COMMENT '更新时间',
  `remark` varchar(255) DEFAULT NULL COMMENT '备注',
  `id` int NOT NULL AUTO_INCREMENT COMMENT '分类ID',
  `category_name` varchar(255) NOT NULL COMMENT '分类名称',
  `category_code` varchar(255) NOT NULL COMMENT '分类编码',
  `parent_id` int DEFAULT NULL COMMENT '父分类ID',
  `category_level` int DEFAULT NULL COMMENT '分类级别',
  `category_status` int DEFAULT NULL COMMENT '分类状态',
  `category_icon` varchar(255) DEFAULT NULL COMMENT '分类图标URL',
  `display_order` int DEFAULT NULL COMMENT '显示顺序',
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COMMENT='商品分类表';

-- 6. 商品信息表：存储商品的详细信息
DROP TABLE IF EXISTS `product_info`;
CREATE TABLE `product_info` (
  `delFlag` int DEFAULT NULL COMMENT '删除标记',
  `createdBy` varchar(32) DEFAULT NULL COMMENT '创建人',
  `createdTime` datetime DEFAULT NULL COMMENT '创建时间',
  `updatedBy` varchar(32) DEFAULT NULL COMMENT '更新人',
  `updatedTime` datetime DEFAULT NULL COMMENT '更新时间',
  `remark` varchar(255) DEFAULT NULL COMMENT '备注',
  `id` int NOT NULL AUTO_INCREMENT COMMENT '商品ID',
  `product_code` varchar(255) NOT NULL COMMENT '商品编码(SKU)',
  `product_name` varchar(255) NOT NULL COMMENT '商品名称',
  `one_category_id` int NOT NULL COMMENT '一级分类ID',
  `two_category_id` int DEFAULT NULL COMMENT '二级分类ID',
  `price` decimal(10,2) NOT NULL COMMENT '商品价格',
  `publish_status` int DEFAULT NULL COMMENT '上下架状态(0下架,1上架)',
  `production_date` datetime NOT NULL COMMENT '生产日期',
  `shelf_life` int DEFAULT NULL COMMENT '保质期(月)',
  `description` varchar(900) DEFAULT NULL COMMENT '商品描述',
  `origin_place` varchar(255) DEFAULT NULL COMMENT '产地',
  `storage_method` varchar(255) DEFAULT NULL COMMENT '存储方法',
  `nutrition_info` varchar(500) DEFAULT NULL COMMENT '营养信息',
  `weight_unit` varchar(50) DEFAULT NULL COMMENT '重量单位',
  `is_organic` tinyint(1) DEFAULT NULL COMMENT '是否有机',
  `is_seasonal` tinyint(1) DEFAULT NULL COMMENT '是否应季',
  `discount_price` decimal(10,2) DEFAULT NULL COMMENT '折扣价',
  `stock` int DEFAULT NULL COMMENT '库存数量',
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COMMENT='商品信息表';

-- 7. 商品图片表：存储商品的图片资源
DROP TABLE IF EXISTS `product_pic`;
```

```sql
CREATE TABLE `product_pic` (
  `delFlag` int DEFAULT NULL COMMENT '删除标记',
  `createdBy` varchar(32) DEFAULT NULL COMMENT '创建人',
  `createdTime` datetime DEFAULT NULL COMMENT '创建时间',
  `updatedBy` varchar(32) DEFAULT NULL COMMENT '更新人',
  `updatedTime` datetime DEFAULT NULL COMMENT '更新时间',
  `remark` varchar(255) DEFAULT NULL COMMENT '备注',
  `id` int NOT NULL AUTO_INCREMENT COMMENT '图片ID',
  `product_id` int NOT NULL COMMENT '所属商品ID',
  `pic_desc` varchar(255) DEFAULT NULL COMMENT '图片描述',
  `pic_url` varchar(255) NOT NULL COMMENT '图片URL',
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COMMENT='商品图片表';

-- 8. 商品库存表：记录商品的库存数量
DROP TABLE IF EXISTS `warehouse_product`;
CREATE TABLE `warehouse_product` (
  `delFlag` int DEFAULT NULL COMMENT '删除标记',
  `createdBy` varchar(32) DEFAULT NULL COMMENT '创建人',
  `createdTime` datetime DEFAULT NULL COMMENT '创建时间',
  `updatedBy` varchar(32) DEFAULT NULL COMMENT '更新人',
  `updatedTime` datetime DEFAULT NULL COMMENT '更新时间',
  `remark` varchar(255) DEFAULT NULL COMMENT '备注',
  `id` int NOT NULL AUTO_INCREMENT COMMENT '库存记录ID',
  `product_id` int NOT NULL COMMENT '商品ID',
  `current_amount` int NOT NULL COMMENT '当前库存数量',
  `min_stock` int DEFAULT NULL COMMENT '最低库存',
  `storage_temperature` varchar(50) DEFAULT NULL COMMENT '存储温度',
  `shelf_life_days` int DEFAULT NULL COMMENT '保质期(天)',
  `batch_number` varchar(100) DEFAULT NULL COMMENT '批次号',
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COMMENT='商品库存表';

-- 9. 购物车表：存储用户加入购物车的商品项
DROP TABLE IF EXISTS `order_cart`;
CREATE TABLE `order_cart` (
  `delFlag` int DEFAULT NULL COMMENT '删除标记',
  `createdBy` varchar(32) DEFAULT NULL COMMENT '创建人',
  `createdTime` datetime DEFAULT NULL COMMENT '创建时间',
  `updatedBy` varchar(32) DEFAULT NULL COMMENT '更新人',
  `updatedTime` datetime DEFAULT NULL COMMENT '更新时间',
  `remark` varchar(255) DEFAULT NULL COMMENT '备注',
  `id` int NOT NULL AUTO_INCREMENT COMMENT '购物车ID',
  `product_id` int NOT NULL COMMENT '商品ID',
  `amount` int NOT NULL COMMENT '购买数量',
  `price` decimal(10,2) NOT NULL COMMENT '加入时单价',
  `user_id` int NOT NULL COMMENT '所属用户ID',
  `is_selected` tinyint(1) DEFAULT 0 COMMENT '是否选中',
  `product_name` varchar(255) DEFAULT NULL COMMENT '商品名称快照',
  `product_pic` varchar(255) DEFAULT NULL COMMENT '商品图片快照',
  PRIMARY KEY (`id`)
```

```
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COMMENT='购物车表';

-- 10. 订单表：记录订单主信息
DROP TABLE IF EXISTS `orders`;
CREATE TABLE `orders` (
  `delFlag` int DEFAULT NULL COMMENT '删除标记',
  `createdBy` varchar(32) DEFAULT NULL COMMENT '创建人',
  `createdTime` datetime DEFAULT NULL COMMENT '创建时间',
  `updatedBy` varchar(32) DEFAULT NULL COMMENT '更新人',
  `updatedTime` datetime DEFAULT NULL COMMENT '更新时间',
  `remark` varchar(255) DEFAULT NULL COMMENT '备注',
  `id` int NOT NULL AUTO_INCREMENT COMMENT '订单ID',
  `order_num` varchar(255) NOT NULL COMMENT '订单编号',
  `user_id` int NOT NULL COMMENT '下单用户ID',
  `shipping_user` varchar(255) NOT NULL COMMENT '收货人姓名',
  `address` varchar(255) NOT NULL COMMENT '收货地址',
  `payment_method` int DEFAULT NULL COMMENT '支付方式(1现金/2余额/3网银/4支付宝/5微信等)',
  `order_money` decimal(10,2) NOT NULL COMMENT '订单商品总金额',
  `shipping_money` decimal(10,2) DEFAULT NULL COMMENT '运费金额',
  `discount_money` decimal(10,2) DEFAULT NULL COMMENT '优惠金额',
  `payment_money` decimal(10,2) NOT NULL COMMENT '实际支付金额',
  `pay_time` datetime NOT NULL COMMENT '支付时间',
  `receive_time` datetime DEFAULT NULL COMMENT '收货时间',
  `order_status` int DEFAULT NULL COMMENT '订单状态(0未支付,1已支付待发货,2已发货,3已完成,4已
取消)',
  `payment_transaction_id` varchar(255) DEFAULT NULL COMMENT '支付交易号',
  `expected_delivery_time` datetime DEFAULT NULL COMMENT '预计送达时间',
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COMMENT='订单表';

-- 11. 订单明细表：记录订单包含的商品项
DROP TABLE IF EXISTS `order_detail`;
CREATE TABLE `order_detail` (
  `delFlag` int DEFAULT NULL COMMENT '删除标记',
  `createdBy` varchar(32) DEFAULT NULL COMMENT '创建人',
  `createdTime` datetime DEFAULT NULL COMMENT '创建时间',
  `updatedBy` varchar(32) DEFAULT NULL COMMENT '更新人',
  `updatedTime` datetime DEFAULT NULL COMMENT '更新时间',
  `remark` varchar(255) DEFAULT NULL COMMENT '备注',
  `id` int NOT NULL AUTO_INCREMENT COMMENT '订单明细ID',
  `order_id` int NOT NULL COMMENT '所属订单ID',
  `product_id` int NOT NULL COMMENT '商品ID',
  `product_name` varchar(255) NOT NULL COMMENT '商品名称(下单时冗余)',
  `amount` int NOT NULL COMMENT '购买数量',
  `product_price` decimal(10,2) NOT NULL COMMENT '商品单价(下单时冗余)',
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COMMENT='订单明细表';

-- 12. 设备信息表（新增）：记录用户绑定的设备
DROP TABLE IF EXISTS `device_info`;
CREATE TABLE `device_info` (
```

```
  `delFlag` int DEFAULT NULL COMMENT '删除标记',
  `createdBy` varchar(32) DEFAULT NULL COMMENT '创建人',
  `createdTime` datetime DEFAULT NULL COMMENT '创建时间',
  `updatedBy` varchar(32) DEFAULT NULL COMMENT '更新人',
  `updatedTime` datetime DEFAULT NULL COMMENT '更新时间',
  `remark` varchar(255) DEFAULT NULL COMMENT '备注',
  `id` int NOT NULL AUTO_INCREMENT COMMENT '设备记录ID',
  `sn_code` varchar(255) NOT NULL COMMENT '设备SN码(序列号)',
  `product_id` int NOT NULL COMMENT '对应商品ID',
  `user_id` int NOT NULL COMMENT '绑定用户ID',
  `bind_date` datetime NOT NULL COMMENT '绑定时间',
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COMMENT='设备信息表';

-- 13. 售后申请表（新增）：记录用户的售后服务请求
DROP TABLE IF EXISTS `service_request`;
CREATE TABLE `service_request` (
  `delFlag` int DEFAULT NULL COMMENT '删除标记',
  `createdBy` varchar(32) DEFAULT NULL COMMENT '创建人',
  `createdTime` datetime DEFAULT NULL COMMENT '创建时间',
  `updatedBy` varchar(32) DEFAULT NULL COMMENT '更新人',
  `updatedTime` datetime DEFAULT NULL COMMENT '更新时间',
  `remark` varchar(255) DEFAULT NULL COMMENT '备注',
  `id` int NOT NULL AUTO_INCREMENT COMMENT '申请单ID',
  `user_id` int NOT NULL COMMENT '提交用户ID',
  `device_id` int NOT NULL COMMENT '相关设备ID',
  `request_type` varchar(100) NOT NULL COMMENT '服务类型',
  `description` varchar(500) DEFAULT NULL COMMENT '问题描述',
  `status` int NOT NULL COMMENT '处理状态(0待处理,1处理中,2已完成,3已拒绝)',
  `request_time` datetime NOT NULL COMMENT '申请时间',
  `process_time` datetime DEFAULT NULL COMMENT '处理完成时间',
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COMMENT='售后申请表';
```

以上 SQL 脚本包含了系统所需的完整表结构。原型系统中的**商品评价表、销售统计表、用户行为记录表**等辅助表在本项目中未作拓展，保留结构以备后用（此处略去定义）。数据库设计遵循第三范式，表间通过主外键关联实现，如订单明细通过 `order_id` 关联订单主表，设备信息和售后申请通过 `user_id`、`product_id` 等关联到相应用户和商品，实现各模块数据的有效关联 [9] [10]。

功能模块说明

小米商城系统主要包括**用户、商品、订单、售后服务**和**后台管理**五大功能模块，各模块下设若干子功能 [11]：

用户模块

- **用户注册**：提供新用户注册功能，用户通过前台注册页面提交姓名、昵称、登录名、密码等信息 [12]。系统检查登录名唯一性避免重复注册 [13]。注册成功后，将新用户信息插入 `user_info` 表（密码需经 MD5 等方式加密存储）。

- **用户登录**：已注册用户可通过登录页面提交凭证进行认证。后端在 `user_info` 表中验证登录名和密码，匹配则允许登录，否则给出错误提示 [14]。登录成功后，用户会话建立，可访问个人账户及购物功能。
- **个人信息管理**：登录用户可查看并更新个人资料，例如修改昵称、头像、联系电话等。用户还可以维护**收货地址**信息（在个人中心添加/编辑/删除地址，设置默认地址），对应操作更新 `delivery_address` 表。用户个人信息更新时，同步修改 `user_info` 表相应字段。必要时用户也可执行注销/退出登录操作，销毁会话。

商品模块

- **商品分类浏览**：商城首页和商品列表页按类别展示商品。商品类别来自 `product_category` 表，支持一级/二级分类层级结构 [15]。前端根据分类层次构建菜单，用户点击分类时，后台查询该分类及子分类下的所有商品列表 [16]。例如点击“手机”类别，将获取分类ID为手机及其子类的所有商品信息。
- **商品搜索**：提供按关键词搜索商品的功能。用户在搜索框输入关键字，系统在 `product_info` 表中执行模糊查询商品名称或描述字段 [17]。匹配的商品列表返回前端显示，方便用户快速定位所需商品。
- **商品详情查看**：用户点击具体商品，可查看商品详情页面。后台根据商品ID从 `product_info` 表获取该商品的完整信息，如名称、价格、描述、规格参数等 [18]。同时从 `product_pic` 表加载商品图片列表 [19]。页面展示商品的详细介绍和轮播图等内容，用户可选择加入购物车。

订单模块

- **购物车管理**：登录用户可将商品加入购物车。每次加入时，在 `order_cart` 表中新建一条记录，保存商品ID、名称快照、单价、数量等 [20]。如果同一商品多次加入，可选择累加数量或生成多条记录（实现策略可自行决定）。用户可在购物车页面查看所有已选商品，更新各商品数量或将商品移出购物车。
- **订单提交**：当用户决定购买购物车中的商品时，进入下单流程。在确认订单页面，用户选择收货地址和支付方式等，然后提交订单。后端首先在 `orders` 表中新建订单主记录，生成唯一的订单编号 `order_num`（可结合日期和随机数） [21]。订单主记录包含用户ID、收货人姓名、收货地址、支付方式、各项金额（商品总额、运费、优惠额、实付金额等）以及初始状态（未支付） [22]。随后，为该订单购物车中的每种商品在 `order_detail` 表插入明细记录，包括商品ID、名称、数量、下单时单价等冗余信息 [23]。订单提交成功后，相关的购物车记录应从 `order_cart` 表删除。（注：Demo 系统中支付流程从简，提交订单后可直接视为已支付）
- **订单支付及状态流转**：订单提交后状态为“未支付”（0）。模拟支付完成后，更新 `orders` 表的 `pay_time` 和状态为“已支付待发货”（1） [4]。管理员发货后，可将状态改为“已发货待收货”（2），并记录发货时间（可利用 `updatedTime` 或扩展字段）。用户确认收货后，状态更新为“已完成”（3)。用户也可在支付前或发货前取消订单，触发状态变更为“已取消”（4) [24]。整个状态流转在 `order_status` 字段记录，配合时间字段刻画订单生命周期。
- **订单查询/管理**：用户可在个人中心查看自己的订单列表及详情。通过用户ID查询 `orders` 表获取其所有订单主信息，并关联查询对应的订单明细列表 [23]。前端显示每笔订单的编号、状态、金额、下单时间等概要信息，可点击展开查看商品明细。对于未支付订单，用户可选择继续支付或取消订单；对于已完成订单，可查看并反馈评价（本项目未实装评价表，仅保留结构）。通过订单管理功能，用户可以方便地追踪订单状态，获取物流和售后支持。

售后服务模块

（售后服务模块为本项目新增特色功能） [25]
- **设备绑定**：用户在收到商品（设备）后，可以通过产品序列号（SN码）将设备与个人账户绑定 [4] [26]。在前台“设备绑定”页面，用户输入设备 SN 码和选择对应商品型号（或由系统自动匹配），提交后后台进行校验。本项目简化处理，可将 SN 码与预置的合法序列号比对来验证真实性。若验证通过，在 `device_info` 表中新建记录，将设备SN码、用户ID、商品ID及绑定时间记录其中 [27]。一个设备SN码只能绑定一次，重复绑定需禁止（可通过查询 `device_info` 确保 SN 唯一）。绑定成功后，该设备信息归属当前用户。 - **使用教程查看**：设备绑定完成后，用户可以在个人账户的设备列表中查看已绑定设备。 [26] 针对每个设备，系统可提供该产品的使用指南或教程链接（例如根据 product_id 加载相应的说明页面）。此功能旨在模拟真实商城中购买电子产品后

提供教程支持的场景。实现上可简单跳转到静态的教程页面或产品介绍页面。 - **售后服务申请**：针对已绑定的设备，用户若遇到故障或需要退换货，可在线提交售后申请 [25] [28]。在“申请售后”页面，用户选择设备（或输入SN码识别设备）并填写申请类型（如维修、退货）和问题描述，然后提交申请。后台将申请信息写入 `service_request` 表，状态初始为0=待处理 [10]。申请提交时间 `request_time` 自动记录当前时间。用户可在售后服务列表中查看自己提交的申请及其状态（待处理/处理中/已完成/已拒绝等）。 - **售后进度与完成**：管理员在后台查看所有售后申请（详见后台管理模块），对用户的申请进行处理。例如确定维修方案或审核退货申请，然后更新 `service_request` 表对应记录的状态为“处理中”(1)或“已完成”(2)等，并填写处理完成时间 `process_time` [10]。若申请不符合规定，可将状态置为“已拒绝”(3)。用户可以实时查看申请状态的变化，并在已完成后收到相应的处理结果通知。通过售后服务模块，实现了购买后在线登记维修退换的闭环流程 [29]。

后台管理模块

后台管理模块提供给商城管理员使用，用于维护网站内容和处理用户交易请求，主要功能包括： - **商品及分类管理**：管理员通过后台界面管理商城商品和类别数据。可新增商品类别（插入 `product_category` 表）、编辑或删除现有类别。商品管理支持新增商品（插入 `product_info` 表及上传商品图片到 `product_pic` 表）、编辑商品信息（修改 `product_info` 表对应记录）、上下架商品（修改其发布状态字段）、以及删除商品（标记 `delFlag` 或实际删除）等。通过完善的商品和分类管理，保证前台展示的数据及时更新且结构清晰。 - **订单管理**：管理员可以查看商城内所有订单记录，并对订单进行必要操作。例如查询 `orders` 表获取不同状态的订单列表，筛选待发货的订单并执行发货操作（这可更新订单状态为已发货并记录发货时间）。对于用户取消的订单，管理员确认后可进行关闭处理。订单管理还涉及协调物流信息（本项目未拓展物流表，可直接在订单备注或扩展字段记录物流单号）。通过后台订单管理，商城运营人员可以监督交易全过程，确保订单按时履约。 - **用户管理**：提供对前台用户账户的查看和维护功能。管理员可以查询 `user_info` 表中的用户列表，查看用户注册信息、账户状态等。如果出现违规用户，管理员可通过删除或设置标记的方式停用其账户（例如利用 `delFlag` 字段模拟删除）。此外，管理员也可维护后台工作人员账号（保存在 `staff_info` 表），例如新增管理员、修改管理员密码等操作，以保障后台系统安全。 - **售后申请处理**：售后服务的申请由管理员在后台进行审核处理。管理员进入售后管理页面，可查询所有 `service_request` 记录，按状态筛选待处理的申请 [10]。对于每条申请，查看其详情（用户、设备、描述等），然后联系用户并给出处理方案。在处理完毕后，管理员更新申请状态为已完成或拒绝，并填写处理时间 [30]。此操作通过调用 `Model.java` 对应方法更新 `service_request` 表，实现前台用户提交->后台管理员处理的闭环。

通过以上后台功能，管理员能够维护商城的数据内容并处理交易全流程，确保商城正常运营。 [25]

Model.java 数据库操作接口设计

由于本项目将所有数据库访问集中在 **Model.java** 类中，建议为各功能模块设计以下数据接口方法（伪代码形式），以便业务逻辑调用。每个方法对应一项数据库操作，方法名及参数含义如下：

- **用户相关方法：**
- `boolean addUser(String name, String nickname, String loginName, String password, String email, String phone)` – **新增用户注册。** 将新用户信息插入数据库的用户表。返回值表示操作是否成功。密码应在调用前加密存储。
- `User getUserByLogin(String loginName, String password)` – **验证用户登录。** 根据提供的登录名和密码查询用户表，匹配则返回用户对象，不匹配返回 null。用于登录校验逻辑。
- `boolean updateUserInfo(int userId, String nickname, String email, String phone)` – **更新用户信息。** 修改用户表中指定用户的昵称、邮箱、电话等资料。成功返回 true。
- `List<Address> getAddressList(int userId)` – **查询收货地址列表。** 根据用户ID从收货地址表检索该用户保存的所有地址，返回地址对象列表。
- `boolean addAddress(int userId, String receiverName, String province, String city, String district, String detail, String phone, boolean isDefault)` – **新增收货地址。** 为用户添加一条新

的收货地址记录。若 `isDefault` 为 true，可在插入后先将该用户其他地址的默认标记清零，再设置此地址为默认。

- `boolean deleteAddress(int addressId)` – **删除收货地址。** 根据地址ID删除（或标记删除）收货地址表中的对应记录。

- **商品及分类相关方法：**

- `List<Category> getCategoryList()` – **获取商品分类列表。** 查询商品分类表，返回所有商品类别（可以按层级组装成树状结构）。用于前台分类菜单展示。
- `List<Product> getProductListByCategory(int categoryId)` – **按分类查询商品。** 根据分类ID查询商品信息表，返回该分类（包括子分类）下所有上架商品列表。实现时可支持一级分类查询所有子类商品。
- `Product getProductDetails(int productId)` – **获取商品详情。** 根据商品ID查询商品信息表及相关的图片表，封装并返回商品详细信息（Product 对象包含图片列表等）。
- `List<Product> searchProducts(String keyword)` – **搜索商品。** 在商品信息表中按名称或描述模糊匹配关键字，返回符合条件的商品列表。
- `boolean addProduct(Product product)` – **新增商品。** 将新的商品信息插入商品信息表，并批量插入商品图片表。`Product` 对象包含商品基本信息和图片URL列表等。此方法主要供后台管理员使用。
- `boolean updateProduct(Product product)` – **更新商品。** 根据商品ID更新商品信息表中对应记录（价格、描述、库存等字段）。如有图片更新，同步修改商品图片表。仅管理员调用。

- `boolean deleteProduct(int productId)` – **删除商品。** 删除指定ID的商品记录，可以是逻辑删除（设置 `delFlag` 标记）或物理删除。需同步删除其图片和相关的库存记录等，确保数据一致。

- **购物车及订单相关方法：**

- `boolean addCartItem(int userId, int productId, int quantity)` – **添加商品到购物车。** 插入一条购物车记录，包含用户ID、商品ID、数量和加入时价格等。若该用户购物车已存在相同商品，可选择改为更新数量（视实现需求）。
- `List<CartItem> getCartItems(int userId)` – **获取购物车列表。** 查询购物车表中某用户的所有记录，按加入时间或店铺分类排序，返回购物车项列表用于前端展示。
- `boolean updateCartItem(int cartId, int newQuantity)` – **更新购物车数量。** 修改购物车表中指定记录的商品数量（如用户在购物车页面调整购买数量）。若 `newQuantity` 为0，可转调删除接口。
- `boolean deleteCartItem(int cartId)` – **删除购物车项。** 移除购物车表中指定的商品记录（用户从购物车删除某商品）。
- `Order createOrder(int userId, int addressId, int paymentMethod)` – **创建订单。** 以用户ID为主体生成新订单：读取该用户购物车中选中的商品，计算总金额和优惠，插入订单主表记录（状态初始为未支付），并批量插入订单明细表记录。`addressId` 用于获取收货人信息和地址（可从地址表查询或由前端直接传入收货信息）。返回生成的订单对象（包含订单号和明细列表）。
- `Order getOrderDetails(int orderId)` – **获取订单详情。** 根据订单ID查询订单主表及其关联的明细记录，封装成完整的订单对象返回，包括订单基本信息和商品项列表。用户可用此方法查看自己的订单内容。
- `List<Order> getOrderListByUser(int userId)` – **获取用户订单列表。** 查询订单表中某用户的所有订单记录，按时间排序返回订单概要列表。用于用户个人中心列出历史订单。
- `boolean updateOrderStatus(int orderId, int status)` – **更新订单状态。** 修改订单表中指定订单的状态字段（例如发货后将状态设为2已发货，并记录发货时间）。管理员发货、用户取消等操作都会调用此方法。返回是否更新成功。

- `boolean updateOrderPayment(int orderId, String paymentTransactionId)` – **订单支付完成处理。** 针对某订单更新其支付状态：设置 `order_status` 为1（已支付待发货），记录支付时间和第三方支付流水号等信息。此方法可由支付回调或模拟支付逻辑调用。

- **售后服务相关方法：**

- `boolean bindDevice(int userId, String snCode, int productId)` – **绑定设备。** 在设备信息表插入一条新记录，将某产品序列号与用户关联。需先验证该SN码未被绑定过（可查询 `device_info` 表确认唯一），再插入包含SN码、用户ID、商品ID、绑定日期的数据。返回绑定操作是否成功。
- `List<Device> getDeviceListByUser(int userId)` – **查询已绑定设备列表。** 检索设备信息表中某用户的所有记录，返回设备对象列表（包含设备ID、SN码、绑定时间、关联商品等）。用户可在前端查看自己绑定的设备。
- `boolean addServiceRequest(int userId, int deviceId, String type, String description)` – **提交售后申请。** 向售后申请表插入一条新记录，包含申请用户、设备ID、服务类型、问题描述、提交时间和状态=0等字段。成功返回 true。通常在用户提交表单时调用。
- `List<ServiceRequest> getServiceRequestsByUser(int userId)` – **获取用户售后申请列表**。查询售后申请表中某用户提交的所有申请记录，返回按时间排序的申请列表，以供用户查看每次申请的进度。
- `List<ServiceRequest> getAllServiceRequests(int statusFilter)` – **获取所有售后申请（管理员用）**。管理员查询所有售后申请记录，可按状态过滤（如仅查询待处理的申请）。返回申请列表用于后台管理界面显示。

- `boolean updateServiceRequestStatus(int requestId, int newStatus)` – **更新售后申请状态（管理员用）**。修改指定申请单记录的状态字段，如将某申请标记为“处理中”(1)或“已完成”(2)等，并相应更新处理完成时间（在 newStatus 表示已处理的情况下设置当前时间）。此方法供后台审核操作调用。

- **后台管理相关方法：**

- `List<User> getAllUsers()` – **获取前台用户列表。** 查询用户信息表获取所有注册用户的数据列表，供后台管理员查看用户概况。
- `Staff checkStaffLogin(String loginName, String password)` – **管理员登录验证。** 在员工信息表（`staff_info`）中根据登录名和密码查询管理员账户，用于后台登录认证。返回 Staff 对象表示验证通过。
- `List<Order> getAllOrders()` – **获取所有订单列表。** 管理员调用，查询订单表获取商城所有订单记录，支持后续按状态或用户筛选，用于后台订单管理总览。
- `boolean addCategory(String name, String code, Integer parentId, int level)` – **新增商品分类。** 在商品分类表插入新类别记录，需指定名称、编码、父类别及级别。
- `boolean updateCategory(int categoryId, String name, String code, Integer parentId, int level, int status)` – **更新商品分类。** 修改现有分类记录的信息或状态（例如启用/停用某分类）。
- `boolean deleteCategory(int categoryId)` – **删除商品分类。** 删除指定ID的商品分类（若其存在子分类或商品关联，应先处理关联再删）。

上述方法清单提供了系统各模块涉及的主要数据库操作接口。开发时，可根据实际需要对方法参数和返回类型作适当调整，例如使用实体对象参数封装多个字段，或使用事务同时完成多个相关表操作。通过将所有数据库访问集中在 `Model.java`，调用者只需通过这一接口层即可完成各业务的数据读写，为后续使用 AI 工具自动生成代码或维护提供了便利。

---

[1] [2] README.md
https://github.com/Aftnos/CQUET-JSP/blob/8c9c0d10c79d7f9be8e267a8d84b67efc36981d0/README.md

[3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] [23] [24] [25] [26] [27] [28] [29] [30] 小米商城项目文档.pdf
file://file-UdtmQe64ucM7CQpi1qBHcr