# Android开发设计

# 课程大作业报告

# 目录

**[概要](#_Toc154586208)** [1](#_Toc154586208)

**[需求说明](#_Toc154586209)** [1](#_Toc154586209)

**[概要设计](#_Toc154586210)** [1](#_Toc154586210)

**[Activity模块](#_Toc154586211)** [1](#_Toc154586211)

**[Fragment模块](#_Toc154586212)** [2](#_Toc154586212)

**[Adapter模块](#_Toc154586213)** [3](#_Toc154586213)

**[Database模块](#_Toc154586214)** [4](#_Toc154586214)

**[关键技术](#_Toc154586215)** [5](#_Toc154586215)

**[成果展示和测试](#_Toc154586216)** [6](#_Toc154586216)

**[成果展示](#_Toc154586217)** [6](#_Toc154586217)

**[商品列表页面](#_Toc154586218)** [6](#_Toc154586218)

**[分类页面](#_Toc154586219)** [8](#_Toc154586219)

**[商品详情页面](#_Toc154586220)** [14](#_Toc154586220)

**[用户登录和注册页面](#_Toc154586221)** [15](#_Toc154586221)

**[购物车页面](#_Toc154586222)** [19](#_Toc154586222)

**[测试过程](#_Toc154586223)** [21](#_Toc154586223)

**[商品列表测试](#_Toc154586224)** [21](#_Toc154586224)

**[分类页面测试](#_Toc154586225)** [22](#_Toc154586225)

**[商品详情测试](#_Toc154586226)** [23](#_Toc154586226)

**[用户登录和注册测试](#_Toc154586227)** [23](#_Toc154586227)

**[购物车测试](#_Toc154586228)** [24](#_Toc154586228)

**[心得体会](#_Toc154586229)** [24](#_Toc154586229)

**概要**

在掌握了安卓的Activity生命周期管理、资源高效利用、Intent组件通信机制、布局的多样性和适配性、Fragment的运用、Adapter的设计模式、ListView的优化、SharedPreferences的存储机制以及SQLite数据库操作等核心知识后，我迈向了一个更具挑战性的实践项目——开发一个购物车应用。此项目的意义在于将所学的知识融会贯通，提供一个实战的舞台，以便更好地理解和掌握安卓开发的基本流程和关键技巧，进一步提升编程能力和技术水平。通过这个项目，我更加熟练地运用安卓开发技术，为未来的项目开发奠定坚实的基础。

**需求说明**

购物APP的设计与实现，该APP包含四个页面，分别是首页、分类页、购物车页和个人中心页，用户可以通过底部导航栏在不同页面之间切换。个人中心页提供用户注册和登录的功能，用户注册时需要验证两次密码的一致性，用户登录后才能使用购物车的功能。首页展示了各种商品的列表，用户可以查看商品的图片、名称、价格和种类，也可以点击商品进入详情页或者在搜索框中输入关键词查找商品。分类页按照商品的种类进行分类，商品分为数码、电脑、家电、服饰和美妆五大类，用户可以点击相应的按钮查看某一类别的商品列表，并且可以从列表中进入商品的详情页。在商品详情页，用户可以详细浏览商品的图片、名称和价格，也可以选择将商品添加到购物车中，每种商品只能添加一次。在购物车页面，用户可以查看已经添加的商品的信息，可以调整商品的数量，也可以选择全选或清除，用户可以对购物车中的商品进行结算，结算后购物车将被清空。

**概要设计**

购物APP主要分为activity, fragment, adapter, database四个模块。

**Activity模块**

activity模块分为4个文件，分别是MainActivity, RegisterActivity, LoginActivity, DetailActivity四个activity，它们分别展示主界面，注册界面，登录界面，商品详情界面。

* MainActivity：该模块首先在onCreate方法中初始化底部导航栏和四个页面的Fragment。通过监听导航栏变化，在onCheckedChanged方法中确定目标页面索引。通过switchFragment方法实现页面切换，判断目标页面是否已添加，已添加则直接显示，未添加则添加后显示。在切换时，调用saveData保存当前页面数据，调用refreshData刷新目标页面数据。通过这样的流程，实现了简洁而有效的页面切换和数据管理。
* RegisterActivity：该模块是一个用户注册模块。在用户输入用户名、密码后，通过数据库查询检查用户是否已存在，若不存在则将用户信息插入数据库，并使用SharedPreferences保存用户登录状态。若用户名已存在或密码不一致，会给予相应提示。最终，根据注册成功与否，跳转至登录页面或提示用户相关信息。
* LoginActivity：该模块是一个用户登录模块。用户输入用户名和密码后，通过查询数据库检查用户名和密码是否匹配，若匹配则跳转至主页面，并使用SharedPreferences保存登录状态。若用户名或密码错误，会给予相应提示。最终，根据登录成功与否，跳转至主页面或提示用户相关信息。
* DetailActivity：该模块为商品详情页模块。根据接收到的商品ID，从数据库查询商品详细信息，包括名称、价格、类型和图片。将查询结果展示在相应控件中。提供"添加到购物车"按钮，点击后根据用户登录状态，将商品添加到购物车数据库中，并给予相应提示。

**Fragment模块**

fragment 模块分为5个文件，分别是 BaseFragment、HomeFragment、TypeFragment、CartFragment 和 UserFragment。BaseFragment 充当其他 fragment 的基类，而其他 fragment 分别用于展示首页、分类页、购物车页和个人中心页。

* BaseFragment ：是一个抽象类，作为其他具体 Fragment 的基类。在 onCreate 方法中，获取 Activity 的上下文。在 onCreateView 中调用抽象方法 initView 来初始化视图。onActivityCreated 调用 initData 初始化数据。定义了抽象方法 refreshData 和 saveData 用于刷新数据和保存数据，具体实现由子类完成。这个模块提供了通用的基类结构，为具体的 Fragment 子类提供了一致的生命周期和方法接口。
* HomeFragment ：通过继承 BaseFragment 实现首页的展示，包括搜索框、搜索按键和商品列表。在 initData 方法中，通过数据库查询获取商品数据，并使用 HomeAdapter 在 RecyclerView 中展示。监听搜索框的文本变化，实现动态搜索和显示搜索结果。点击搜索按钮时，执行搜索操作并刷新界面。在 refreshData 中重新加载商品列表。
* TypeFragment 是商品分类展示页面的具体实现，继承自 BaseFragment。在 initView 方法中初始化视图，包括多个商品类别的按钮和商品列表。在 initData 方法中加载所有商品数据，并设置默认的按钮颜色。通过按钮的点击事件，根据不同的商品类别查询数据库，更新商品列表的显示。Listener 方法中为每个按钮添加点击事件监听器，根据点击的按钮更新商品类别，刷新商品列表。在 refreshData 和 saveData 方法中提供具体的实现。
* CartFragment 实现了购物车展示的功能，继承自 BaseFragment。在 initView 方法中初始化购物车页面的视图，包括商品列表、全选按钮、清空购物车按钮、总价显示和结算按钮。在 initData 方法中加载购物车中的商品数据，并设置购物车适配器。通过全选按钮的点击事件，更新购物车中商品的选中状态，计算总价并显示。清空购物车按钮点击事件则清空购物车中的所有商品。结算按钮点击事件会删除购物车中的商品并提示用户总花费。
* UserFragment 作为用户界面的展示模块，继承自 BaseFragment。在 initView 方法中初始化用户页面的视图，包括登录按钮、注册按钮和显示欢迎信息的 TextView。通过 SharedPreferences 获取用户登录信息，显示在欢迎信息中。在 Listener 方法中，设置登录按钮和注册按钮的点击事件，分别跳转到登录页面和注册页面。initData 方法用于初始化用户数据，包括设置点击事件监听器。

**Adapter模块**

Adapter模块分为3个文件，分别是HomeAdapter, TypeAdapter, CartAdapter三个Adapter，它们分别实现首页商品列表的展示，分类页商品列表的展示，购物车商品列表的展示。

* HomeAdapter 是用于展示首页商品列表的适配器。通过继承 RecyclerView.Adapter，实现了视图绑定和数据加载的功能。HomeViewHolder 类用于封装商品列表项的视图，包括商品图片、商品名称、商品价格和商品类型。在构造方法中设置了点击事件，点击列表项会跳转到商品详情页面。onCreateViewHolder 方法用于创建并返回 HomeViewHolder 对象，该方法会在需要新的列表项视图时调用。onBindViewHolder 方法用于将数据绑定到 HomeViewHolder 对象，根据 Cursor 中的数据设置商品的名称、价格、类型和图片。同时，设置点击事件，点击列表项会启动 DetailActivity 并传递商品的 id。getItemCount 方法返回列表项的数量，即 Cursor 对象的行数，表示首页商品的数量。
* TypeAdapter 是用于展示分类页商品列表的适配器。通过继承 RecyclerView.Adapter，实现了视图绑定和数据加载的功能。TypeViewHolder 类用于封装商品列表项的视图，包括商品图片、商品名称、商品价格和商品类型。在构造方法中设置了点击事件，点击列表项会跳转到商品详情页面。onCreateViewHolder 方法用于创建并返回 TypeViewHolder 对象，该方法会在需要新的列表项视图时调用。onBindViewHolder 方法用于将数据绑定到 TypeViewHolder 对象，根据 Cursor 中的数据设置商品的名称、价格、类型和图片。同时，设置点击事件，点击列表项会启动 DetailActivity 并传递商品的 id。getItemCount 方法返回列表项的数量，即 Cursor 对象的行数，表示分类页商品的数量。
* CartAdapter 用于展示购物车商品列表的适配器。通过继承 RecyclerView.Adapter，实现了视图绑定和数据加载的功能。CartViewHolder 类用于封装购物车商品列表项的视图，包括商品图片、商品名称、商品价格、商品类型、商品数量、减少商品数量按钮和增加商品数量按钮。在构造方法中设置了点击事件和按钮点击事件，点击列表项会跳转到商品详情页面，点击按钮会减少或增加商品数量，并更新数据库中对应的数量。onCreateViewHolder 方法用于创建并返回 CartViewHolder 对象，该方法会在需要新的列表项视图时调用。onBindViewHolder 方法用于将数据绑定到 CartViewHolder 对象，根据 Cursor 中的数据设置商品的名称、价格、类型、图片和数量。同时，设置按钮的点击事件。getItemCount 方法返回列表项的数量，即 Cursor 对象的行数，表示购物车商品的数量。getTotalPrice 方法计算购物车中所有商品的总价，遍历 Cursor 对象，获取每个商品的价格和数量，计算商品的小计，并累加到总价中，最后返回总价。

**Database模块**

database模块分为3个文件，分别是ProductDatabaseHelper, UserDatabaseHelper, CartDatabaseHelper. 它们分别实现商品数据库的管理，用户数据库的管理，购物车数据库的管理。

* ProductDatabaseHelper 是一个用于管理商品数据库的辅助类。在 onCreate 方法中，创建名为 "Product" 的数据库表，表包括商品的 id、name、price、type、image 字段。然后插入了20个商品的初始数据，包括商品的名称、价格、类型和图片。在 onUpgrade 方法中，如果数据库表 "Product" 存在，将其删除并重新创建。整体流程包括数据库的创建和升级，以及初始商品数据的插入。
* UserDatabaseHelper 是一个用于管理用户数据库的辅助类。在 onCreate 方法中，创建名为 "users" 的数据库表，表包括用户的 id、usname（用户名）、uspwd（密码）字段。在 onUpgrade 方法中，如果数据库表 "users" 存在，将其删除并重新创建。整体流程包括数据库的创建和升级。该类用于管理用户信息的存储和访问。
* CartDatabaseHelper 是一个用于管理购物车数据库的辅助类。在 onCreate 方法中，创建名为 "Cart" 的数据库表，表包括购物车项的 id、name（商品名）、price（价格）、type（种类）、image（图片）、count（商品个数）、username（用户名）字段。insertCart 方法用于向购物车中插入商品信息，queryCart 方法用于查询购物车中是否存在特定商品，updateCart 方法用于更新购物车中商品的数量，deleteCart 方法用于删除购物车中的商品信息。整体流程包括数据库的创建以及购物车商品的增删改查。

**关键技术**

1. SQLite 数据库管理: 在应用开发过程中，我利用了 SQLite 数据库进行本地数据存储和检索，以确保购物应用的数据管理得以完善。在数据库设计中，我规划了购物车数据库与用户数据库、商品数据库之间的关系。具体而言，我将购物车数据库的表项设置为商品数据库的表项，并额外添加了用户名和商品数量字段。这种设计使得购物车数据库的管理更为灵活，通过关联用户名，我能够实现对购物车的个性化操作，包括根据用户进行增加、删除、修改和查询商品。这种结构既简洁又高效，为购物车的数据管理提供了可靠的基础。这一巧妙的关系设计不仅确保了数据的一致性，同时也提升了用户体验，使得购物车操作更加智能和符合个性化需求。
2. RecyclerView 的使用: 在项目中，我运用了 RecyclerView 实现了灵活的列表展示，其中包括了三个不同适配器的创建（HomeAdapter、TypeAdapter 和 CartAdapter），以及对列表项点击事件的处理。在一开始，我面临了有效利用 RecyclerView 展示商品列表的挑战，尤其是在如何将从数据库获取的数据传递给 RecyclerView 时感到困扰。通过深入学习 Cursor 的使用方法，我逐渐熟悉了从数据库结果中提取数据的技巧，并运用 ViewHolder 将数据动态展示在 RecyclerView 中。这个过程不仅提升了我的数据处理能力，也让我更加熟练地使用 RecyclerView 来呈现不同类型的商品列表。
3. Fragment 的生命周期管理: 在项目中，我深入理解了 Fragment 的生命周期管理，通过继承 BaseFragment 并在其中巧妙地管理生命周期方法，实现了不同界面片段的适时初始化和数据刷新。对于一个设计精美的应用而言，合理利用 Fragment 是至关重要的。一开始，我直接让各个页面继承自 Fragment，但这样做增加了代码的冗余，需要在每个 Fragment 中分别实现各自的生命周期方法，包括 onCreate、onCreateView 等，并且还需要实现各自的数据刷新或保存方法。为了解决这一问题，我构建了抽象类 BaseFragment。通过BaseFragment，我为不同的 Fragment 子类提供了统一的生命周期和方法接口，使得它们共享通用的基类结构。这样的设计不仅降低了代码的重复度，还提高了代码的可维护性和可扩展性。
4. SharedPreferences 的使用: 在项目中，我利用了 SharedPreferences 来进行简单的数据存储，特别是在用户登录和注册功能的实现中。通过 SharedPreferences，我成功地存储了用户的登录信息，主要是用户名。这一策略为购物车模块的设计提供了便捷的方式，使得在用户访问购物车时，能够根据存储的用户名来精准地访问购物车数据库。在用户登录注册流程中，我使用 SharedPreferences 存储了用户的用户名，这样无论用户何时访问购物车，系统都可以根据存储的用户名迅速定位到对应的购物车数据。
5. Intent 的传递和接收: 运用 Intent，我成功实现了在不同活动之间的数据传递，尤其是从商品列表跳转到商品详情页的场景。通过 Intent，我将商品的唯一标识（商品id）传递到商品详情页，为用户提供了流畅的商品查看体验。在商品列表页跳转到商品详情页的过程中，我使用 Intent 携带商品id信息，这样商品详情页就能准确地根据id从商品数据库中检索并展示相应的商品信息。

**成果展示和测试**

**成果展示**

**商品列表页面**



首页展示了多种商品，包括名称、价格和图片。用户可以通过滚动列表查看所有商品。



首页点击搜索框输入搜索内容后点击搜索后可以根据商品名字搜索商品。

**分类页面**



在分类页点击全部按键可查看全部商品。



在分类页点击数码可查看数码类别商品。



分类页点击电脑按键可以查看电脑类别商品。



分类页点击家电按键可以查看家电类商品。



分类页点击服饰按键可以查看服饰类商品。



在分类页点击美妆按键可以查看美妆类商品。

**商品详情页面**



用户点击商品列表中的商品后进入。显示商品的详细信息，包括名称、价格、类型和图片。提供“加入购物车”按钮，允许用户将商品添加到购物车。

**用户登录和注册页面**



个人中心页用户提供用户注册与登录。



在个人中心页点击注册按键，用户可以输入用户名、密码、确认密码来进行注册。

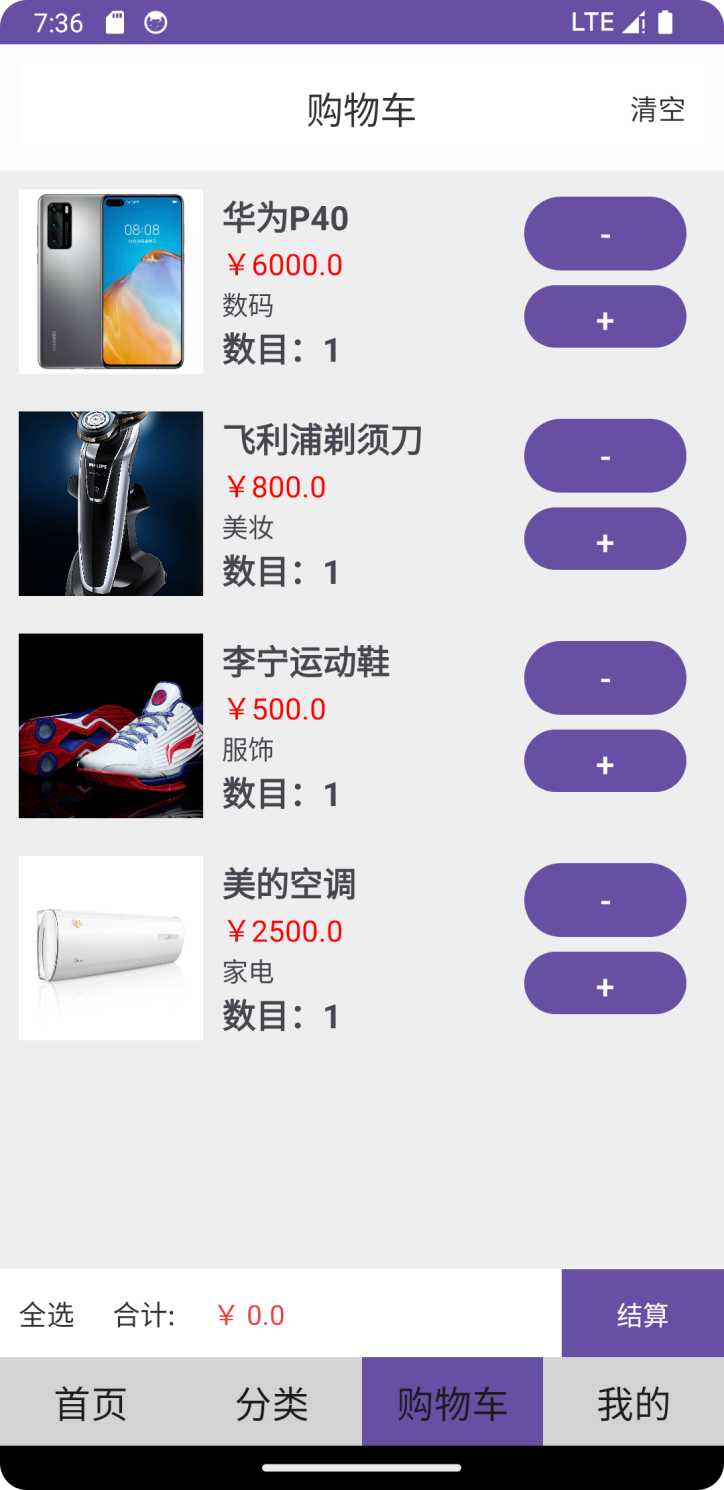


在个人中心页点击登录按键，用户可以输入用户名和密码来进行登录。

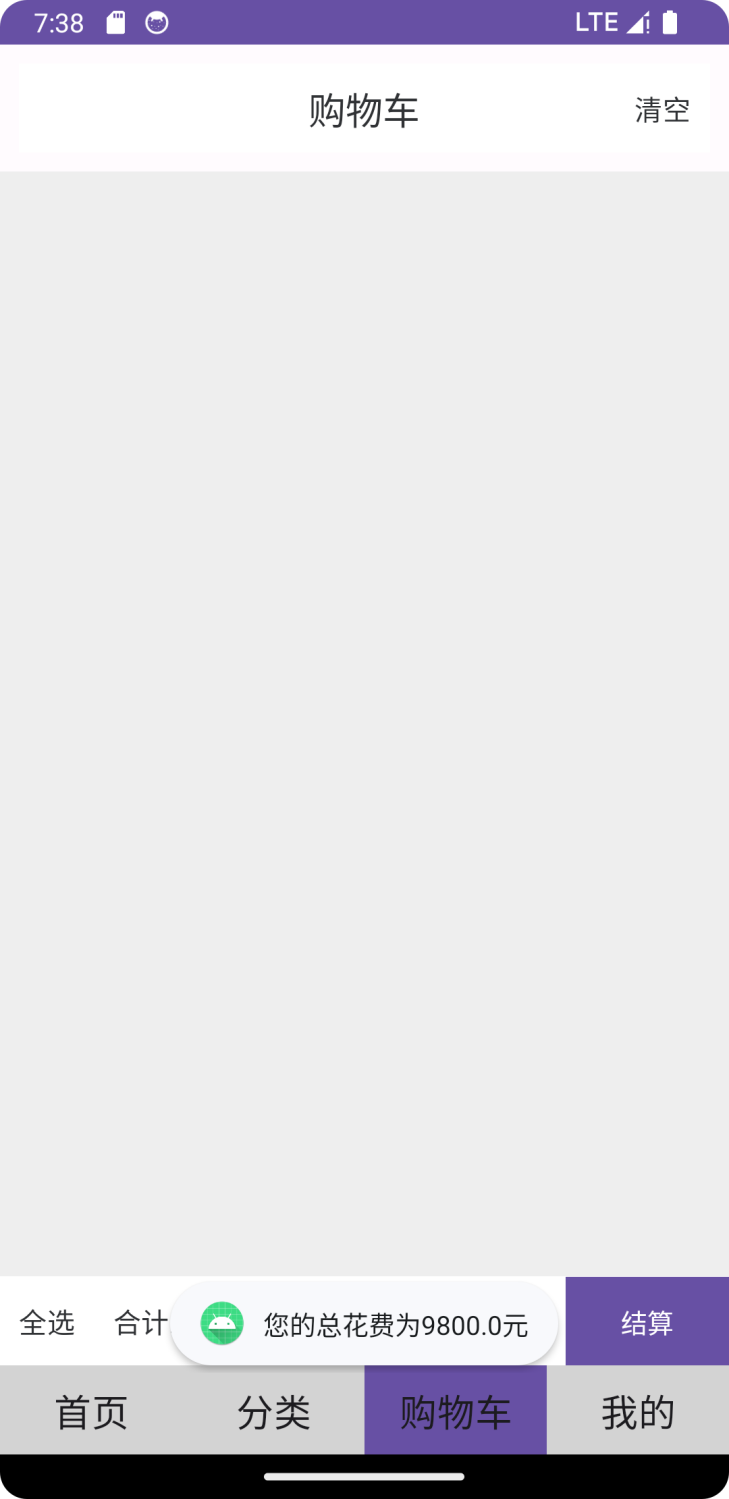


登录成功后个人中心页展示欢迎，显示当前用户名。

**购物车页面**



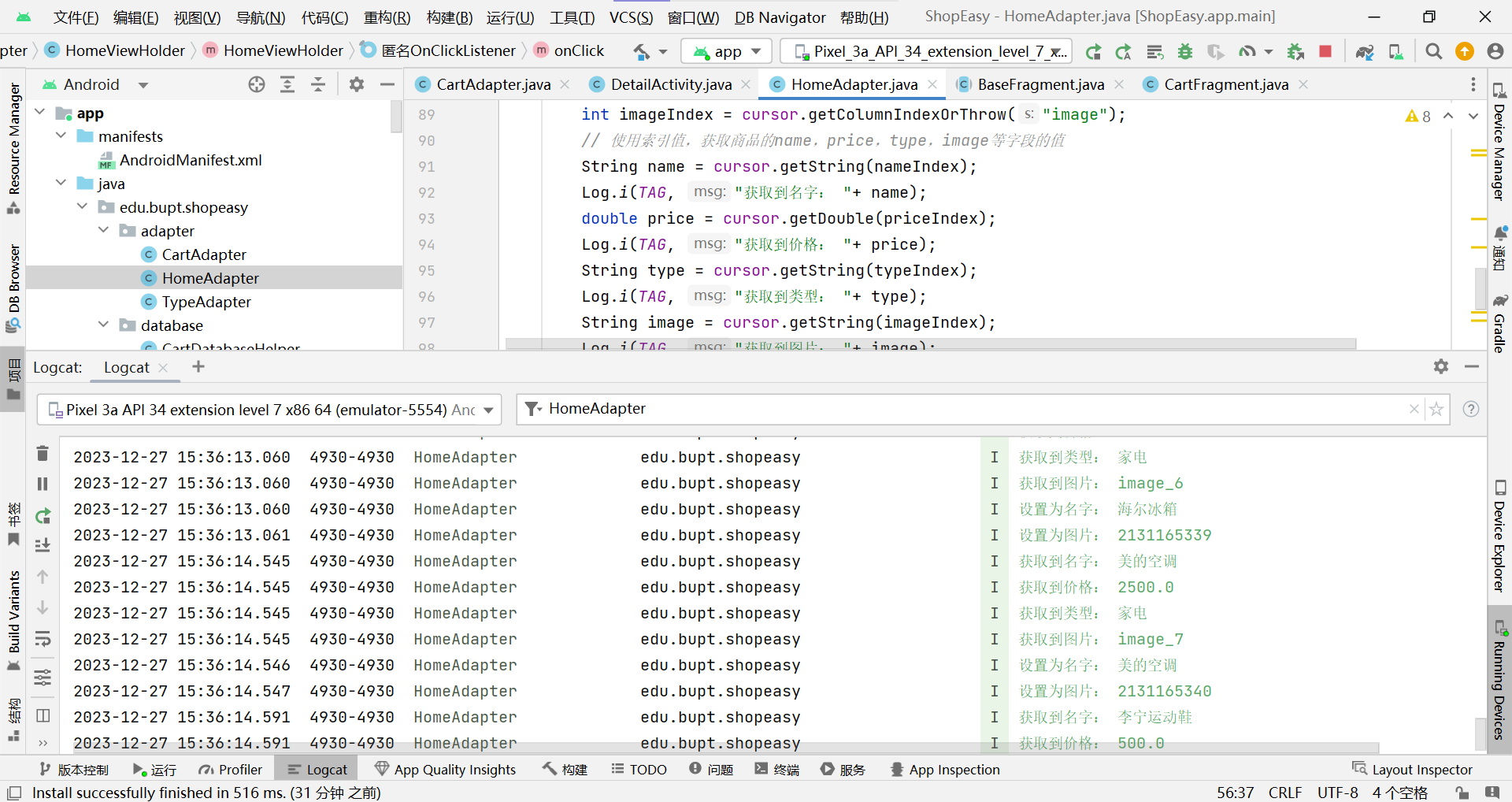
购物车页面列出用户已添加到购物车的商品。显示每个商品的名称、数量、价格。提供增加和减少商品数量的按钮。用户可以清空或结算购物车。



在购物车页面点击全选，再点击结算可以对购物车进行结算，提示用户总花费。

**测试过程**

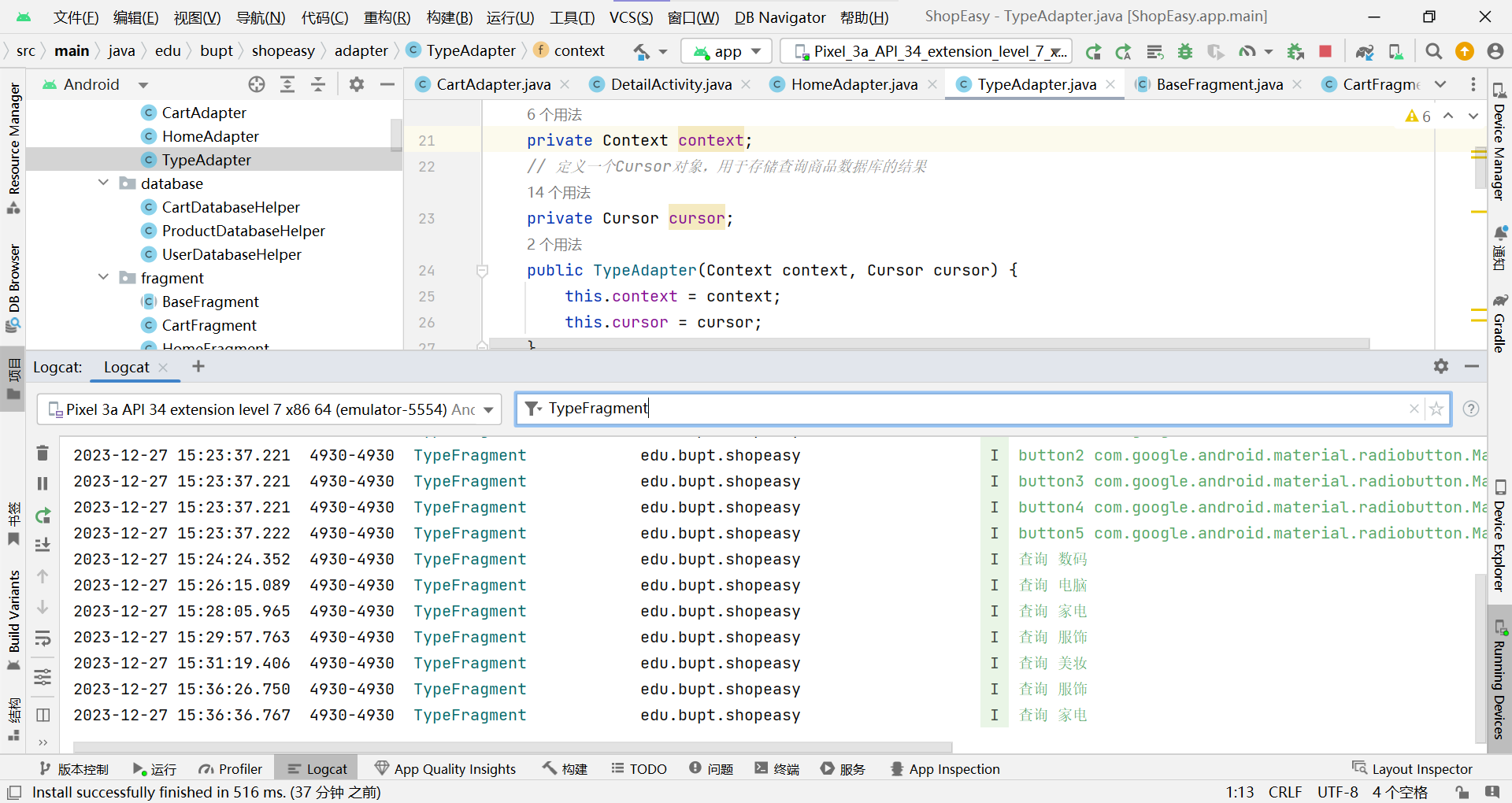
**商品列表测试**



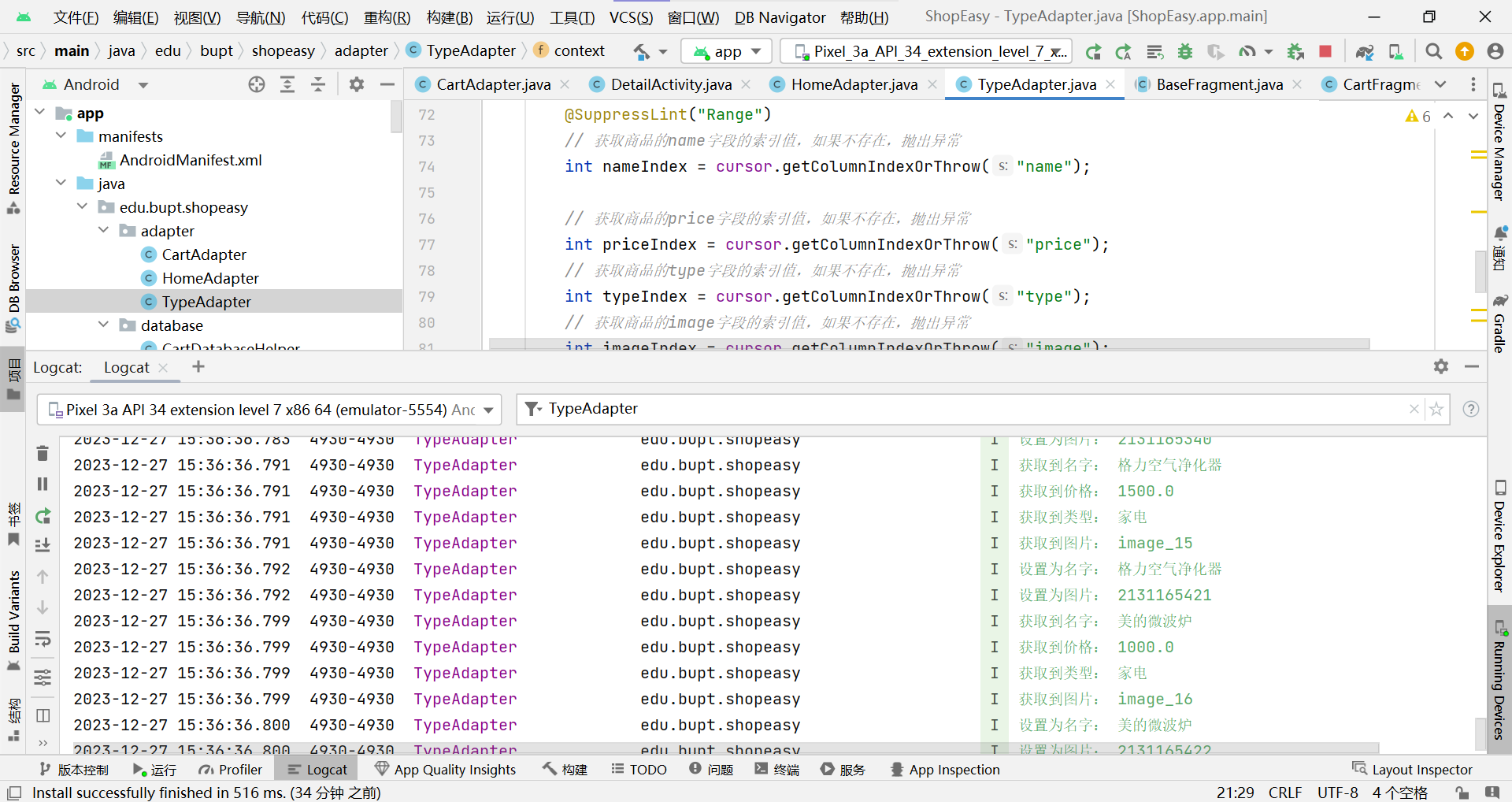
打开应用，检查商品列表是否正确显示。检查滚动列表是否能够正常浏览。在HomeAdapter调试输出中可以看到正确获取到数据库数据并显示在商品列表上。

**分类页面测试**

打开应用，查看商品分类页是否正确显示，点击各种分类按键查看分类展示是否正常。



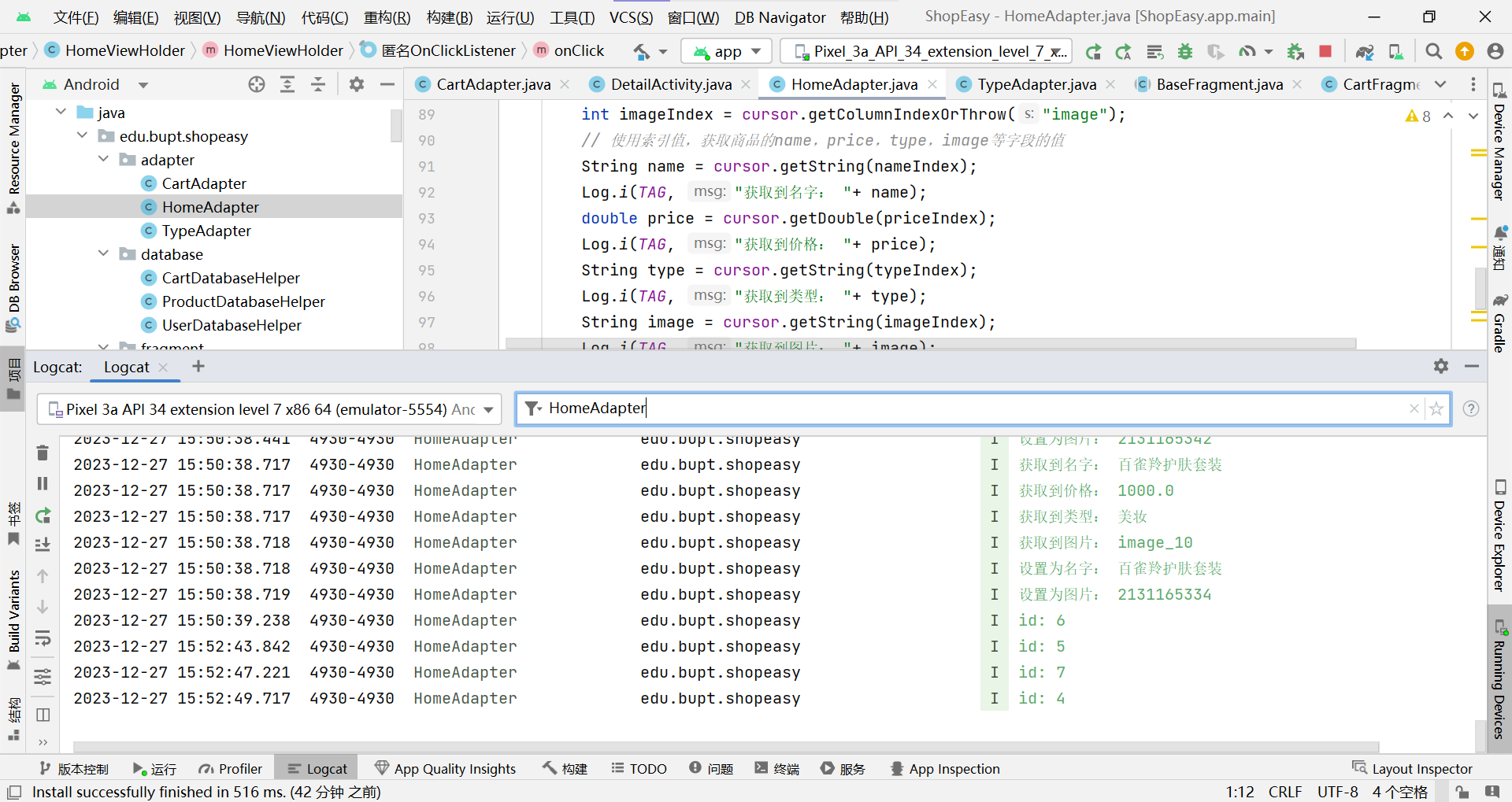
在TypeFragment调试输出可以看到分类查询的正确响应。



在TypeAdapter调试输出可以看到正确获取到了商品列表并将其展示出来。

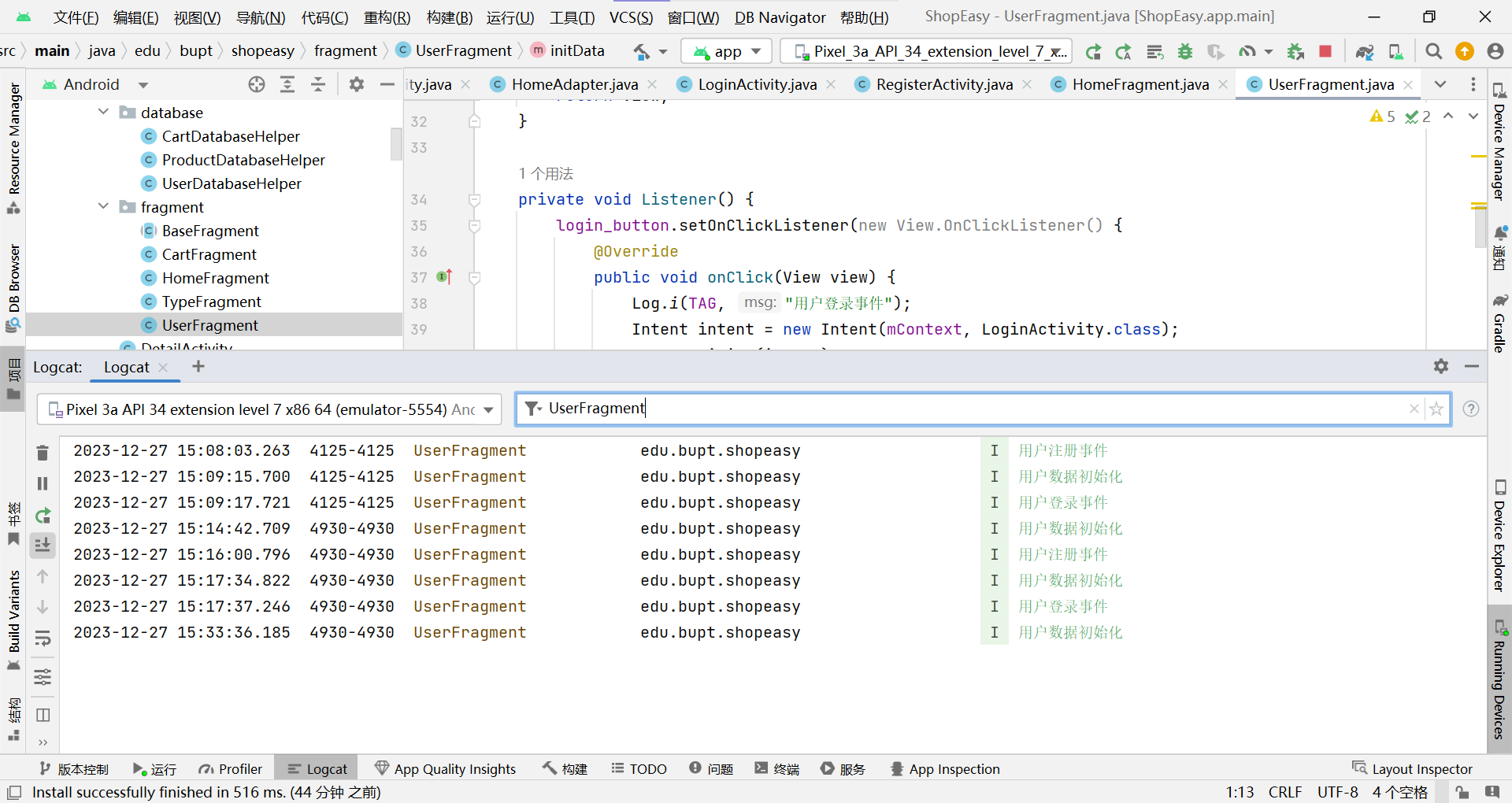
**商品详情测试**

从商品列表点击进入商品详情页面。确保商品详细信息准确显示。测试“加入购物车”按钮是否成功将商品添加到购物车。



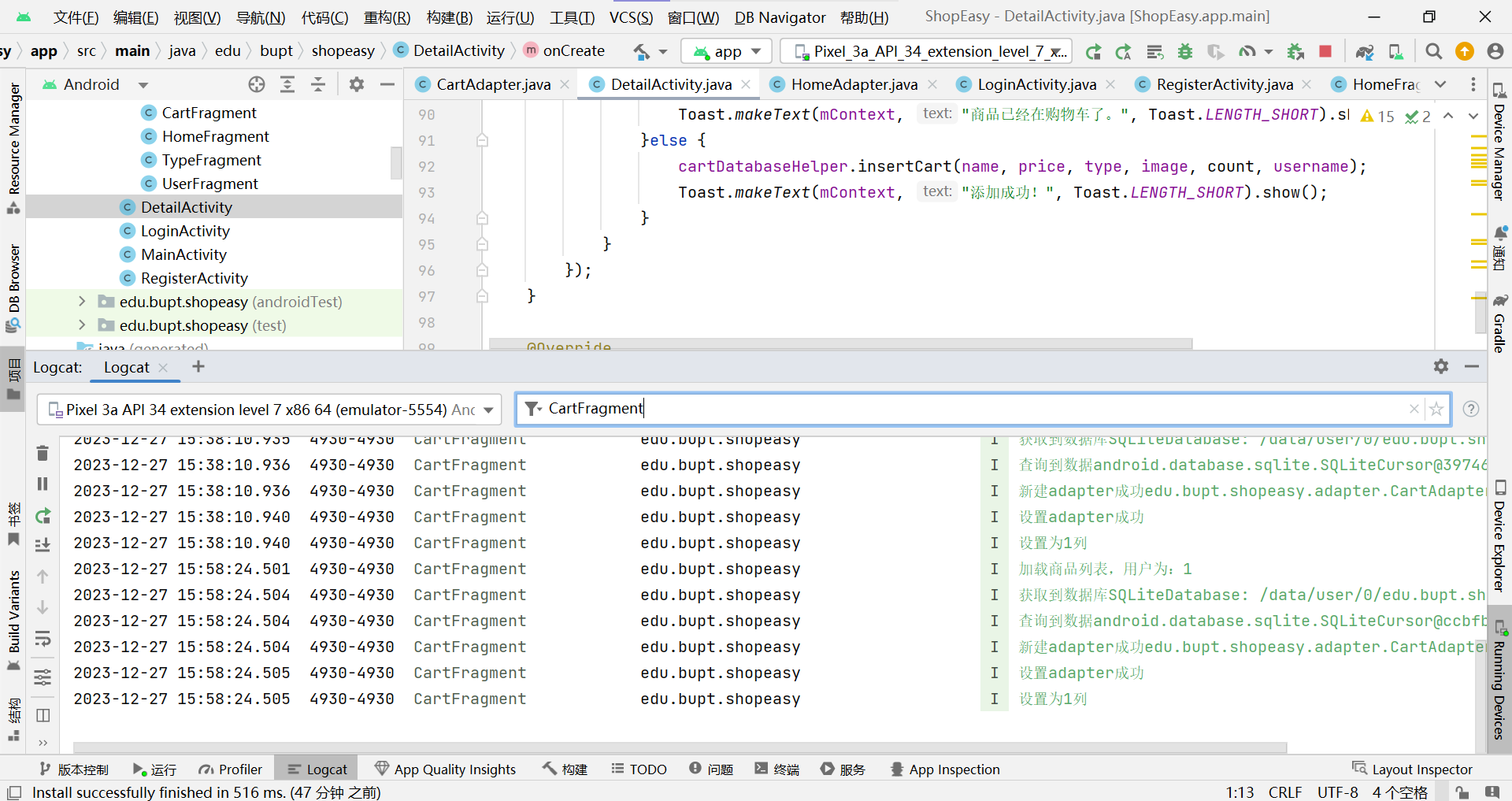
在HomeAdapter调试输出可以看到用户点击的商品的id，确保用户点击id与商品详情页展示出的商品内容相符合。

**用户登录和注册测试**



测试用户登录和注册功能，确保用户身份验证正常工作。在UserFragment的调试输出可以看到用户的注册登录事件。

**购物车测试**



从商品详情页面将商品添加到购物车。进入购物车页面，确保已添加的商品显示正确。测试增加和减少商品数量的按钮是否正常工作。在HomeAdapter的调试输出中可以看到当前加载的购物车的具体信息。如上图，该购物车属于用户名为1的用户，且被成设置为1列来展示。

**心得体会**

在这个学期的大作业中，我设计并实现了一个购物APP，这是一个非常有趣和有用的项目，让我学到了很多Android开发的知识和技能。在这个项目中，我使用了SQLite数据库来存储和检索用户、商品和购物车的数据，使用了RecyclerView来实现灵活的列表展示，使用了Fragment来实现不同界面片段的适时初始化和数据刷新，使用了SharedPreferences来存储用户的登录信息，使用了Intent来实现不同活动之间的数据传递和接收。这些技术让我能够实现一个功能完善、性能优良、用户友好的购物APP。

在这个项目的过程中，我也遇到了一些挑战和困难，例如如何设计数据库的表结构，如何从数据库结果中提取数据，如何使用ViewHolder将数据动态展示在RecyclerView中，如何利用BaseFragment来管理Fragment的生命周期，如何利用Intent携带商品id信息等。为了解决这些问题，我查阅了很多资料和文档，观看了很多教程和视频，参考了很多示例和代码，通过不断的尝试和调试，最终成功地完成了我的项目。这个过程让我感受到了Android开发的魅力和乐趣，也提高了我的数据处理能力、界面设计能力、逻辑思维能力和问题解决能力。

对于我的项目，我还有一些改进和完善的建议，以提高我的APP的竞争力和吸引力。我想加入一些用户评价和反馈的功能，让用户可以对商品和服务进行评价和评论，这样可以增加用户的信任和参与感，也可以帮助商家改进商品和服务的质量。我也想加入一些推荐和优惠的功能，利用数据分析和机器学习的技术，根据用户的喜好和行为，向用户推荐一些合适的商品和优惠信息，这样可以增加用户的购买意愿和满意度，也可以提高商家的销售额和利润。我还想加入一些社交和分享的功能，让用户可以和其他用户交流和互动，分享自己的购物心得和体验，这样可以增加用户的忠诚度和活跃度，也可以扩大我的APP的影响力和知名度。