Software engineering 1 - ReMarket

Mateusz Osik
Mateusz Polis
Mateusz Nędzi
Sebastian Rydz
Filip Shramko

December 2024

Contents

1 Introduction

Purpose of the Document

This document serves as the final system specification for ReMarket, a dynamic platform designed to facilitate the buying and selling of second-hand goods. By focusing on sustainability and affordability, ReMarket a',s to promote the reuse of items, reducing waste and fostering a circular economy. The specification consolidates and refines the functional, structural, and business requirements of the system, providing a comprehensive guide for stakeholders.

Scope of the System

ReMarket is a web-based platform that connects individuals and businesses interested in trading pre-owned goods. It emphasizes an intuitive user experience and robust functionality to support diverse user needs, including:

- Listing and browsing second-hand items.
- Facilitating secure transactions between buyers and sellers.
- Providing tools for managing user profiles, listings, and transaction histories.

The system is designed to handle high traffic volumes, ensure data security, and support seamless scalability as the user base grows. Key stakeholders include individual users, small businesses, and administrators responsible for maintaining platform operations.

2 Class Diagram Description

The class diagram in Figure 1 provides an overview of the main entities in the ReMarket software system and the relationships between them. This diagram is critical for understanding how the components of the system interact to facilitate functionalities such as listing items, managing accounts, tracking orders, and processing payments. Below, we describe each class and its dependencies:

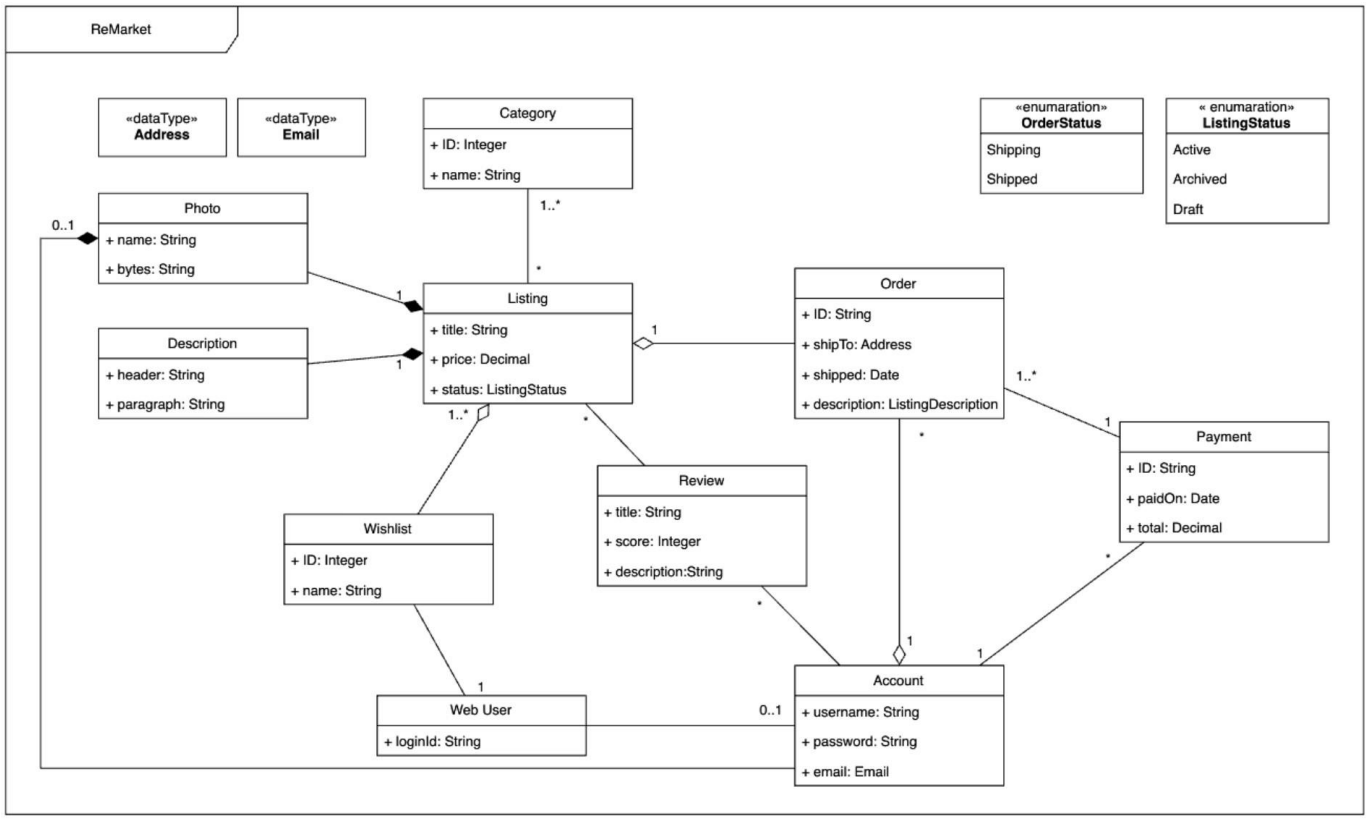


Figure 1: ReMarket Class Diagram

2.1 Core Classes

Account: The Account class represents the user account details. It contains attributes such as username, password, and email, information about suspension, as well as role in the system (admin, user), which are essential for user authentication and communication. Each account is associated with one or more Web User entities, representing the system's flexibility to manage multiple web users.

Web User: The Web User class is an abstraction for users interacting with the system. It is associated with a unique loginId and has a one-to-one relationship with the Wishlist, which enables users to save listings for future consideration.

Wishlist: The Wishlist class allows users to manage a collection of desired items. Each wishlist has a unique ID and name and is directly linked to the Listing class, indicating that the saved items must correspond to active listings.

Listing: The Listing class is central to the marketplace, representing items posted by sellers. Key attributes include title, price, and status (using the ListingStatus enumeration, which defines states like Active, Archived, and Draft). Each listing belongs to a specific Category, has a Description, and can include multiple Photo objects to showcase the item visually.

Photo: The Photo class provides additional details for each listing by storing images. It contains attributes like name and bytes, representing the image data. A listing can have zero or more photos (0..*), reflecting optional visual content for listings.

Description: The Description class provides textual information for listings, with attributes such as header and paragraph. Each listing is required to have one description (1), ensuring that sufficient information about the item is always available.

Category: The Category class organizes listings into logical groupings. It includes an ID and a name, enabling the system to classify and filter listings. Each category can have multiple associated listings (1..*), ensuring scalability for diverse item categories.
Review: The Review class allows buyers to provide feedback on listings, helping to build trust within the platform. It includes attributes such as title, score (e.g., a numerical rating), and description for detailed comments. Reviews are linked directly to listings, showing that reviews pertain to specific items.

2.2 Order and Payment System

Order: The Order class manages the transactional aspect of the platform. Attributes include ID, shipTo (an Address object), shipped (a date when sent), and description (which uses the ListingDescription (description as string and an array of photos (files)) data type). Each order is associated with one or more listings (1..*), indicating that orders can encompass multiple items.
Payment: The Payment class represents the financial component of the system. It includes attributes such as ID, paidOn (a date), and total (a decimal value representing the payment amount). Payments are directly linked to orders, ensuring that each order has an associated payment.
Address: The Address data type is utilized by the Order class to specify the shipping destination for purchased items. It ensures that orders are associated with precise delivery locations.

2.3 Enumerations and Data Types

ListingStatus: The ListingStatus enumeration defines the possible states for a listing, such as Active, Archived, and Draft. This helps sellers manage their inventory efficiently by specifying the visibility of their listings.
OrderStatus: The OrderStatus enumeration categorizes orders into states like Shipping and Shipped. This ensures clarity in the order lifecycle and allows buyers to track progress effectively.
Email: The Email data type ensures standardization and validation of email addresses for both user accounts and communications within the platform.
Dependencies and Relationships: - The Account class is foundational for authentication, connecting to Web User entities for user-specific actions. - The Listing class is central, linked to several entities like Category, Photo, and Description to provide comprehensive item details. - The Order and Payment classes are tightly coupled, managing the transaction and payment processes. - The Wishlist and Review classes enhance user experience by enabling personalization and trust-building features.

This class diagram ensures a modular and extensible design, supporting key functionalities of the ReMarket platform while maintaining clear relationships between entities.

3 Requirements

3.0 Web Customer Stories

Web Customer stories focus on the fundamental interactions that users can have with the platform, such as registering for an account and logging in. These functionalities ensure a smooth onboarding process and secure access to userspecific features.

3.0.1 User Registration

User registration is the foundational feature that enables new users to join the platform and access personalized features. By creating an account, users can save their preferences (light/dark mode, cookies, etc.), manage their wishlist, and view their order history. This process ensures a secure and user-friendly onboarding experience.
Priority: Must have
User Story: As a new user, I want to create an account by providing my personal information, so that I can save my preferences, wishlist, and order history for future visits.

Acceptance Criteria:

- The registration form must require the following fields:
  - Name
  - Email
  - Password
- The system must validate the email format before allowing registration.
- Passwords must meet the following complexity requirements:
  - Minimum 8 characters
  - At least one uppercase letter
  - At least one lowercase letter
  - At least one number
- The registration form must provide immediate feedback for invalid inputs (e.g., incorrect email format, weak passwords).
- Upon successful registration, the system must send a confirmation email to the user.

Non-Functional Requirements:

- The registration process must complete within 2 seconds.
- The system must support up to 1000 simultaneous registrations without performance degradation.
- All user data, particularly passwords, must be securely stored using encryption techniques such as bcrypt.

3.0.2 Logging In

The logging in functionality allows returning users to securely access their accounts. This is critical for managing their saved data, such as wishlists and ensuring continuity in their user experience.
Priority: Must have
User Story: As a returning user, I want to log in using my email and password, so that I can access my account, including my saved wishlist and shopping cart.

Acceptance Criteria:

- The login page should prompt for an email and password.
- Incorrect login attempts should display an error message without exposing account details.
- Logged-in users should be redirected to their account dashboard.

Non-Functional Requirements:

- The login authentication must complete within 1 second.
- The system must support at least 500 concurrent logins.
- Login endpoints must be protected against brute force attacks using rate limiting.

3.1 Buyer Stories

This section outlines the key functionalities available to buyers on the platform. These stories focus on enabling buyers to browse, search, evaluate, and manage items, as well as track orders and interact with sellers. These features ensure a seamless purchasing experience and build trust within the platform.

3.1.1 View Listings

Viewing listings is a fundamental feature for buyers, allowing them to browse the available items in the marketplace. This functionality ensures that buyers can explore all options and make informed purchase decisions.
Priority: Must have
User Story: As a buyer, I want to see all listings created by sellers, so that I can choose an item to buy.

Acceptance Criteria:

- Listings are displayed with the following basic information:
  - Title
  - Price
  - Category
  - Thumbnail image
- Listings are paginated to support large volumes.
- Listings can be sorted by criteria such as:
  - Price
  - Newest
  - Popularity

Non-Functional Requirements:

- Listings should load in under 2 seconds, and images should render clearly at thumbnail size.
- Listings are always up-to-date, with a 99.9% uptime for data availability.
- The system should support concurrent users with no noticeable slowdowns.
- Only approved listings are displayed.

3.1.2 Search Functionality

The search functionality provides buyers with an efficient way to find specific products using keywords and filters. This ensures that users can quickly locate items of interest, even in large inventories.
Priority: Should have
User Story: As a buyer, I want to search for items using keywords, so that I can quickly find specific products.

Acceptance Criteria:

- Search results should match keywords in the title and description.

- Search results are returned in under 2 seconds for typical queries.
- Search filters can be applied, such as:
  - Category
  - Price range

Non-Functional Requirements:

- The search bar is visible and easy to use.
- Search results accurately reflect available listings.
- The system handles multiple concurrent searches without delay.
- The system prevents SQL injection and other vulnerabilities.

3.1.3 Browsing by Category

Browsing by category allows buyers to explore items in a structured and intuitive way. Categories and subcategories make it easier to navigate the marketplace and discover relevant products.
Priority: Should have
User Story: As a buyer, I want to browse items by category, so that I can easily find products I'm interested in.

Acceptance Criteria:

- Items are grouped by category and subcategory.
- Users can switch between categories with minimal load time.
- Pagination within each category works smoothly.

Non-Functional Requirements:

- Categories are easy to navigate and structured hierarchically.
- Categories are consistently displayed, with no missing items.
- Loading time under 2 seconds.

3.1.4 Viewing Item Details

The item details page provides buyers with all the necessary information to make informed purchase decisions. By displaying high-resolution images, detailed descriptions, and seller information, this feature enhances buyer confidence and decision-making. Priority: Should have
User Story: As a buyer, I want to view detailed information and images of an item, so that I can make an informed purchase decision.

Acceptance Criteria:

- The item page displays the following information:
  - Title

  - Description
  - Price
  - Images
  - Seller information (name, photo)
- Users can view multiple images of an item in high resolution.
- Related items or suggestions are provided.

Non-Functional Requirements:

- Item details load within 1 second.
- Accurate and complete details are shown.
- High-resolution images load progressively.
- Images and details are protected from unauthorized access.

3.1.5 Wishlist Functionality

The wishlist functionality enables buyers to save items for future consideration. This feature is valuable for users who want to track their favorite products and return to them later.
Priority: Could have
User Story: As a buyer, I want to add items to a wishlist, so that I can save them for future consideration.

Acceptance Criteria:

- Users can add or remove items from their wishlist.
- Wishlisted items persist across sessions.
- Users can view and manage their wishlist in a dedicated section.

Non-Functional Requirements:

- Wishlist management is accessible from multiple pages.
- Wishlist persists across device logins.
- Adding or removing items takes under 1 second.
- Wishlist access is restricted to authenticated users.

3.1.6 See Seller Ratings and Reviews

Ratings and reviews build trust between buyers and sellers by providing transparent feedback about past transactions. This feature helps buyers make confident decisions when choosing sellers.
Priority: Could have
User Story: As a buyer, I want to see seller ratings and reviews, so that I can know who I'm buying from.

Acceptance Criteria:

- User profiles display aggregate ratings and recent reviews.
- Reviews are sortable by relevance or date.

Non-Functional Requirements:

- Ratings and reviews are easy to find and readable.
- Review information updates accurately after purchases.
- Ratings load without delay on seller profile.
- Prevent manipulation of ratings/reviews.

3.1.7 Track Orders

Order tracking allows buyers to monitor the status of their purchases, providing clear information about shipping and delivery times. This feature enhances the post-purchase experience and builds trust in the platform.
Priority: Could have
User Story: As a buyer, I want to track my orders, so that I know when to expect delivery.

Acceptance Criteria:

- Orders display current status (e.g., “Shipped”, “Delivered”).
- Estimated delivery date is provided.
- Users receive notifications for order status updates.

Non-Functional Requirements:

- Order status is easy to find and understand.
- Status updates are accurate and timely.
- Status changes appear instantly upon update.
- Only the buyer can view their full order details.

3.1.8 Leave Reviews for Sellers

Leaving reviews allows buyers to share their experiences with others, promoting accountability and transparency. This feature helps maintain the platform’s reputation and fosters trust among users.
Priority: Could have
User Story: As a buyer, I want to leave reviews for sellers, so that I can share my experience with others.

Acceptance Criteria:

- Buyers can submit a rating and optional review after delivery.
- Reviews are visible to other users.
- Reviews are moderated for inappropriate content.

Non-Functional Requirements:

- Review submission process is clear and simple.
- Reviews are posted without errors.
- Review posting occurs instantly.
- Reviews are only permitted from verified buyers.

3.2 Seller Stories

This section focuses on the functionalities that enable sellers to list items, manage their inventory, and interact with buyers. These features are essential for creating a dynamic marketplace and ensuring that sellers can effectively showcase their products.

3.2.1 List Items for Sale

Listing items is the core functionality for sellers, allowing them to add products to the marketplace. This feature ensures that sellers can provide accurate and appealing information about their items.
Priority: Must have
User Story: As a seller, I want to list items for sale, so that I can reach potential buyers.

Acceptance Criteria:

- Sellers can create a listing with title, description, price, and images.
- Listings appear in the marketplace upon creation.
- Sellers can set availability (e.g., in-stock/out-of-stock).

Non-Functional Requirements:

- Listing creation is intuitive and quick (low number of steps and views).
- Listings appear consistently to all users.
- Listings publish instantly.
- Only authorized sellers can list items.

3.2.2 Upload Multiple Photos

Uploading multiple photos enables sellers to visually showcase their products, helping buyers understand the condition and appearance of items. This feature is critical for building buyer confidence.
Priority: Should have
User Story: As a seller, I want to upload multiple photos of my item and choose a thumbnail, so that buyers can see its condition.

Acceptance Criteria:

- Sellers can upload multiple images in high resolution and choose a thumbnail.
- Images appear correctly in listing details.

- Sellers can reorder or delete images.

Non-Functional Requirements:

- Image upload is streamlined and allows batch uploads.
- Images load reliably for buyers.
- Uploads complete in under 2 seconds per image.
- Images are protected from unauthorized downloads.

3.2.3 Set the Price

Price-setting gives sellers control over their sales and ensures that they can adjust their pricing strategy as needed. This feature is fundamental to the seller's ability to manage their inventory.
Priority: Must have
User Story: As a seller, I want to set the price, so that I have control over the sale.

Acceptance Criteria:

- Sellers can set and adjust the price at any time.
- Price changes update immediately on the listing.
- Buyers are notified if a price changes while the item is in their wishlist.

Non-Functional Requirements:

- Price-setting interface is straightforward.
- Price changes apply consistently.
- Price updates reflect in under 1 second.
- Only sellers can modify prices of their items.

3.2.4 Edit or Delete Listings

Editing or deleting listings allows sellers to keep their inventory up-to-date and accurate. This ensures that buyers only see relevant and available products.
Priority: Should have
User Story: As a seller, I want to edit or delete my listings, so that I can keep my inventory up-to-date.

Acceptance Criteria:

- Sellers can edit details (title, description, price, images, categories) of their listings.
- Sellers can mark listings as sold or delete them.
- Deleted or marked-sold listings no longer appear in buyer searches.

Non-Functional Requirements:

- Editing and deletion are accessible within the seller's dashboard.
- Changes propagate across all buyer views instantly.
- Edits apply in under 1 second.
- Only the seller of a listing can edit or delete it.

3.3 Admin Stories

This section outlines the administrative functionalities that ensure the platform operates smoothly and securely. Admins manage user accounts, monitor listings, and prevent fraudulent activity, maintaining the integrity of the marketplace.

3.3.1 Manage User Accounts

Managing user accounts allows admins to maintain a safe and compliant marketplace environment. This feature enables admins to enforce platform rules and address violations.
Priority: Must have
User Story: As an admin, I want to manage user accounts, so that I can maintain a safe marketplace environment.

Acceptance Criteria:

- Admins can view, suspend, or delete user accounts.
- Admin actions on accounts are logged.
- Suspended users cannot log in or make transactions.

Non-Functional Requirements:

- Admin interface for account management is efficient and easy to use.
- Account changes apply instantly.
- Admin actions take effect in real-time.
- Only admins can modify user accounts.

3.3.2 Review and Remove Inappropriate Listings

This feature allows admins to ensure that listings comply with platform policies. By removing inappropriate content, admins maintain the quality and trustworthiness of the marketplace.
Priority: Must have
User Story: As an admin, I want to review and remove inappropriate listings, so that the platform complies with policies.

Acceptance Criteria:

- Admins can flag, edit, or delete listings that violate policies.
- Listings flagged for review appear in an admin dashboard.
- Notifications are sent to sellers if a listing is removed.

Non-Functional Requirements:

- Flagging and removal actions are efficient.
- Actions taken on listings apply immediately.
- Listings update in real-time after admin action.
- Only authorized admins can review and remove listings.

3.3.3 Monitor Transactions for Suspicious Activity

Monitoring transactions enables admins to detect and address fraudulent behavior, protecting the platform and its users from malicious activities.
Priority: Should have
User Story: As an admin, I want to monitor transactions for suspicious activity, so that I can prevent fraudulent behavior.

Acceptance Criteria:

- Admins receive alerts for flagged transactions (by external payment provider).
- Admins can view transaction history and associated user details.
- Admins can place holds on suspicious transactions (after external payment provider flags the first try).

Non-Functional Requirements:

- Alerts for suspicious transactions are clearly identifiable.
- System flags transactions accurately based on rules.
- Alerts generate in under 1 second from the flagged action.
- Only authorized admins have access to transaction details.

3.4 Use Case Diagram Description

The following subsections describe the primary use case diagrams for the ReMarket platform, detailing the actors, use cases, and their relationships.

3.4.1 Use Case Diagram for Existing Web Customer

The use case diagram in Figure 2 illustrates the interactions between the primary actors (Seller, Buyer, and Payment Provider) and the key functionalities of the ReMarket platform. This diagram provides a high-level view of the system's capabilities and how different actors utilize these functionalities. Below, we describe the roles and their respective use cases in detail.

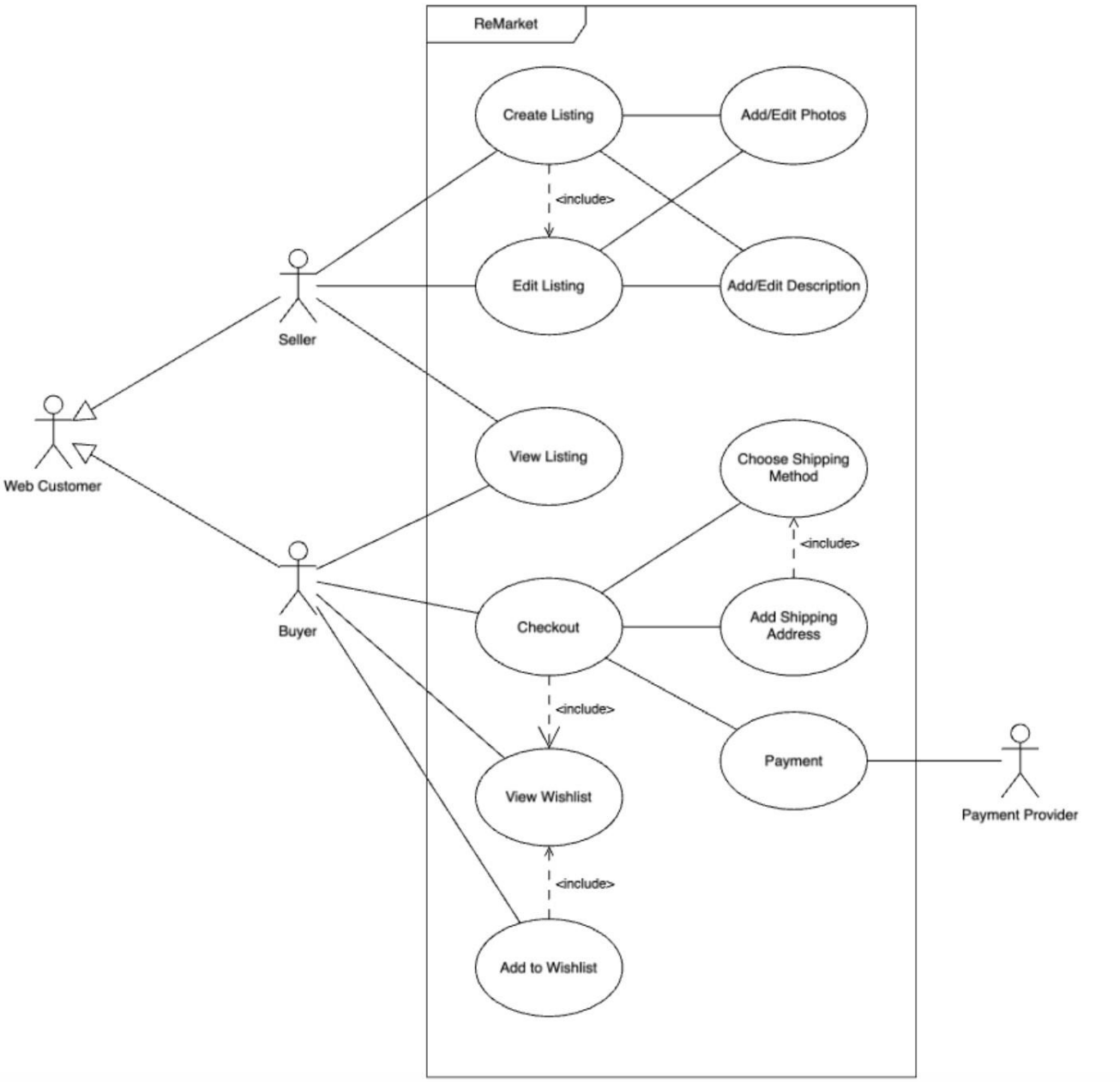


Figure 2: Use Case Diagram for Existing Web Customer Actors in the System

Web Customer: The Web Customer is a generalization of the Seller and Buyer actors. This means that every seller or buyer has access to core platform functionalities common to all users, such as creating an account, logging in, and viewing available listings. The Web Customer does not directly perform specialized actions like creating listings or managing purchases but serves as a base user type.

Seller: The Seller is an actor who lists items for sale on the platform. Sellers are responsible for creating and managing their listings, which includes adding or editing photos and descriptions. Sellers ensure their items are accurately represented and categorized to reach potential buyers effectively.
Buyer: The Buyer is an actor who interacts with the platform to browse and purchase items. Buyers can view listings, add items to their wishlist, and proceed to checkout. They are also responsible for specifying shipping details and completing payments to finalize transactions.
Payment Provider: The Payment Provider is an external system or service responsible for processing payments securely. This actor interacts with the platform during the checkout process to ensure transactions are executed correctly and securely.

Use Cases and Their Relationships

Create Listing: This use case allows sellers to add new items to the marketplace. Sellers can include important details such as photos, descriptions, and prices. The Create Listing use case includes two additional use cases: -
Add/Edit Photos: Enables sellers to upload and modify images to provide visual details for their listings. - Add/Edit Description: Allows sellers to input or update textual information about the listed item.
Edit Listing: Sellers can update existing listings to ensure accurate information or reflect changes in availability or price. This use case leverages the same sub-features as Create Listing, including editing photos and descriptions. View Listing: Both buyers and sellers can view details about items in the marketplace. Buyers use this functionality to evaluate potential purchases, while sellers use it to verify the presentation of their listings.
Checkout: This use case allows buyers to purchase items from the platform. The Checkout process includes the following related use cases: - Choose Shipping Method: Buyers can select their preferred shipping option based on cost, delivery speed, or other factors. - Add Shipping Address: Buyers provide the destination address for item delivery, ensuring accurate order fulfillment. - Payment: This use case is linked to the Payment Provider actor, which securely processes the transaction and ensures that sellers receive payment.
View Wishlist: Buyers can access their wishlist to review previously saved items. This use case allows users to revisit items of interest for future consideration. It also includes the Add to Wishlist use case, which enables buyers to save items from the marketplace to their wishlist.

Use Case Dependencies

The diagram demonstrates several include relationships, where one use case incorporates the functionality of another. For example: - Create Listing includes Add/Edit Photos and Add/Edit Description, as these actions are integral to creating a comprehensive listing. - Checkout includes Choose Shipping Method, Add Shipping Address, and Payment, as these are mandatory steps for completing a transaction.
The use of include relationships ensures modular and reusable design, allowing related functionalities to be encapsulated and reused across different use cases.

System Coverage

This diagram effectively highlights the core capabilities of the ReMarket platform. It ensures that the needs of both buyers and sellers are met through essential functionalities like listing management, transaction processing, and wishlist management. Additionally, the inclusion of the Payment Provider actor reflects the platform's reliance on external systems for secure financial transactions.
By clearly defining the roles of each actor and their interactions with the system, the use case diagram ensures a comprehensive understanding of the platform's functionality and its support for different user workflows.

3.4.2 Use Case Diagram for Admin Panel

The use case diagram in Figure 3 depicts the interactions between the Admin, Staff, and external services (ListingService and AccountService) within the AdminPanel subsystem of the ReMarket platform. This diagram focuses on administrative functionalities, including managing user accounts and listings, ensuring policy compliance, and maintaining platform safety.

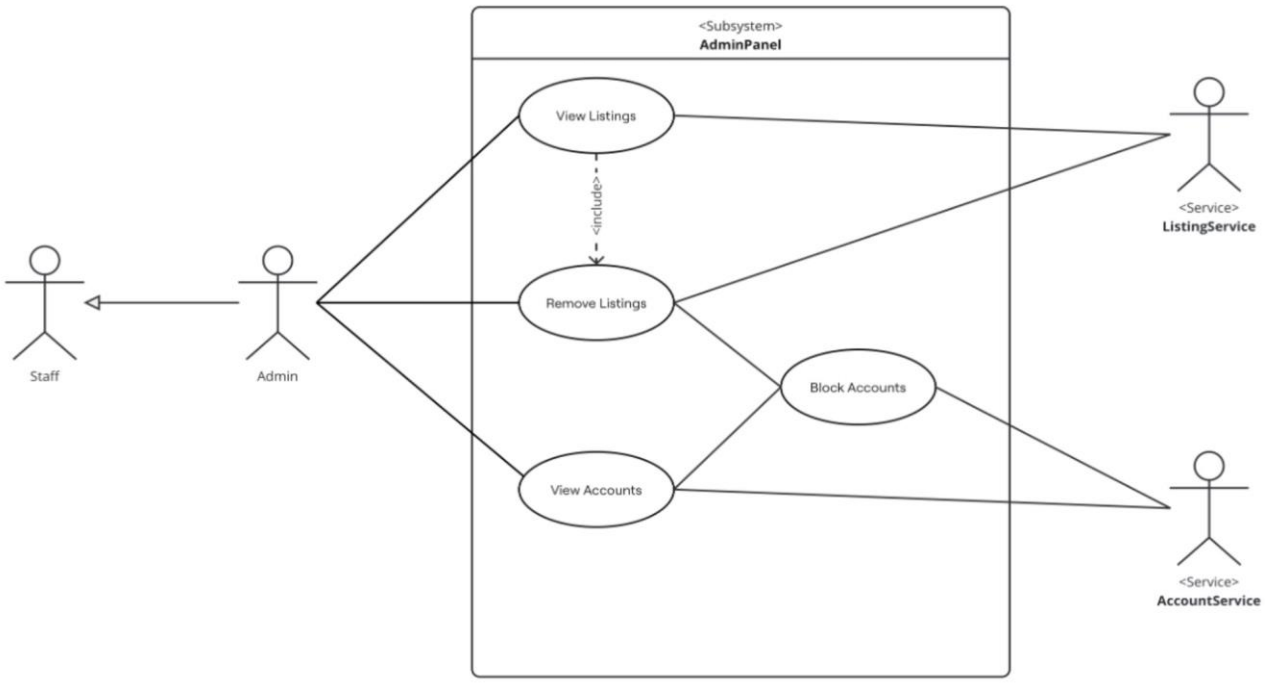


Figure 3: Use Case Diagram for Admin Panel

Actors in the System

Admin: The Admin actor is responsible for managing the ReMarket platform. Admins have elevated privileges to perform actions such as removing inappropriate listings, blocking user accounts, and viewing sensitive user or listing data. The Admin interacts directly with the AdminPanel subsystem.
Staff: The Staff actor represents additional personnel who assist the admin in performing specific tasks. For example, staff may have access to view listings or accounts but require admin approval for critical actions such as account blocking or listing removal.

ListingService: The ListingService is an external service that provides data related to active listings on the platform. This service is utilized by the AdminPanel to fetch or update information when performing actions like removing listings.
AccountService: The AccountService is an external service that manages user account information. The AdminPanel interacts with this service to view accounts, update account statuses, or block users when necessary.

Use Cases and Their Relationships

View Listings: This use case allows admins and staff to access the details of all active listings. It is a foundational use case that provides the context for other administrative actions, such as removing listings. The Remove Listings use case includes View Listings as a prerequisite step, ensuring that the admin evaluates a listing before deciding to remove it.

Remove Listings: The Remove Listings use case enables admins to delete or archive listings that violate platform policies. This action directly interacts with the ListingService to ensure that the removal is reflected across the platform.
View Accounts: Admins and staff can use this functionality to access user account details, such as account history, user actions, or reported issues. This use case is critical for monitoring and maintaining a safe environment on the platform.
Block/Ban Accounts: The Block/Ban Accounts use case allows admins to restrict access for users who violate platform rules. Blocking a user interacts with the AccountService to enforce the restriction across all aspects of the platform. This use case is often initiated after reviewing account details in the View Accounts use case.

Subsystem Overview and Dependencies

The AdminPanel subsystem serves as the interface for admin-related functionalities. It encapsulates the use cases required to manage listings and user accounts effectively. The subsystem depends heavily on the external ListingService and AccountService for fetching and updating data, ensuring that the admin actions are accurately reflected in the platform's main system.
Include Relationships: The diagram uses include relationships to show dependencies between use cases. For example: - The Remove Listings use case includes View Listings, as viewing a listing is a necessary step before deciding to remove it. - Similarly, reviewing accounts through View Accounts is often a prerequisite for the Block Accounts use case.

System Coverage

This diagram highlights the key administrative functionalities needed to maintain the platform's safety, enforce policies, and manage listings and accounts effectively. By clearly delineating the roles of the admin, staff, and external services, the diagram ensures a robust understanding of the platform's administrative operations and their dependencies on external systems.

3.4.3 Use Case Diagram for New vs Registered Customers

The use case diagram in Figure 4 illustrates the differences in actions that can be taken by New Customers and Registered Customers within the ReMarket platform. Both actors are specialized roles of the Web Customer actor and have access to distinct functionalities based on their registration status.

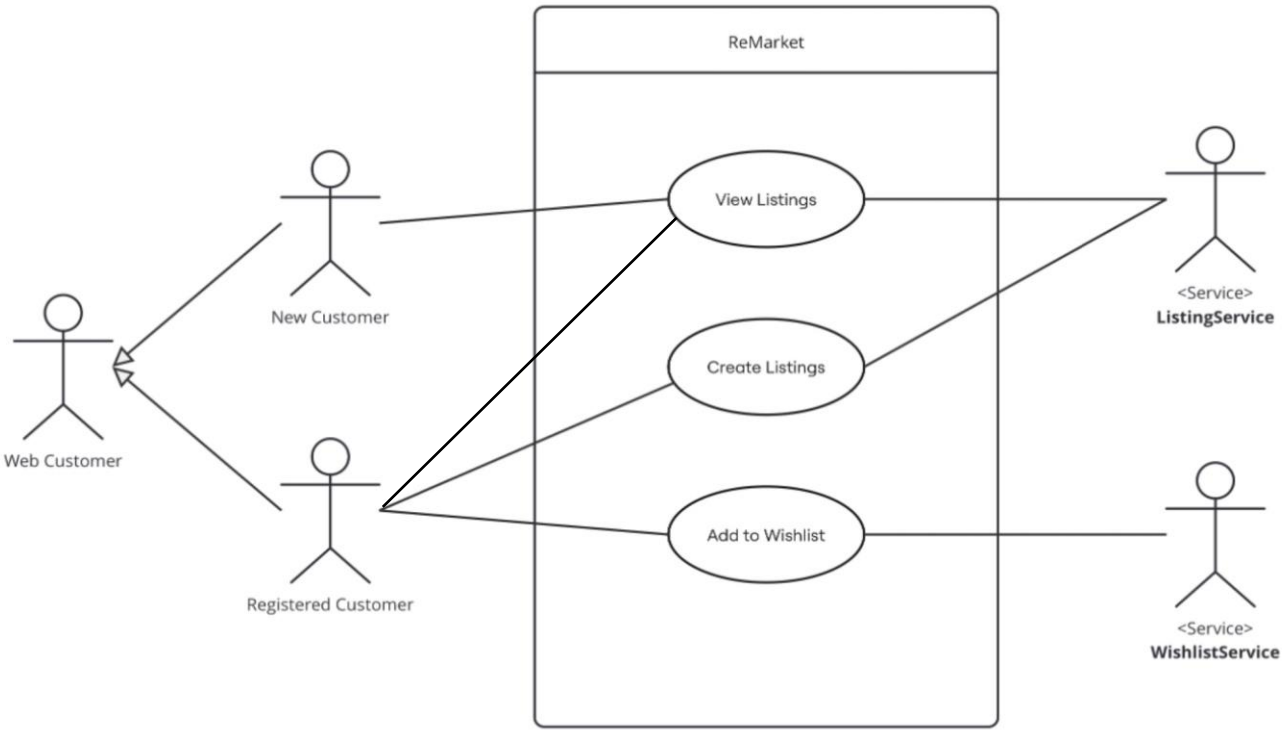


Figure 4: Use Case Diagram for New vs Registered Customers

Actors in the System

Web Customer: The Web Customer is the generalized actor representing all users interacting with the platform. Both New Customers and Registered Customers inherit from this actor and share the ability to view listings. However, their available actions differ based on their registration status.

New Customer: The New Customer actor represents users who have not yet registered an account on the platform. These users are limited to browsing listings, providing them with an opportunity to explore the marketplace before committing to registration.

Registered Customer: The Registered Customer actor represents users who have created an account on the platform. In addition to the functionality available to new customers, registered/logged-in customers can create their own listings and add items to their wishlist, leveraging the platform's personalized and transactional features.

ListingService: The ListingService is an external service that handles the creation, retrieval, and management of listings. It interacts with both New Customers and Registered Customers during their interactions with listings. WishlistService: The WishlistService is an external service that allows registered customers to save items of interest to their wishlist for future consideration. This service is only accessible to registered customers.

Use Cases and Their Relationships

View Listings: This use case is shared by both New Customers and Registered Customers, allowing them to browse and evaluate items available in the marketplace. The View Listings functionality interacts with the ListingService to fetch and display listing data.

Create Listings: This use case is exclusive to Registered Customers. It enables them to add items to the marketplace by creating detailed listings. The Create Listings use case interacts with the ListingService to store the new listings and make them available to other users.
Add to Wishlist: This use case is also exclusive to Registered Customers. It allows them to save items to their wishlist for future consideration. The Add to Wishlist use case interacts with the WishlistService, ensuring that the saved items persist across sessions and devices.

System Coverage

The diagram emphasizes the distinction between new and registered customers, showcasing how registration unlocks additional functionalities. This differentiation ensures that potential users can explore the platform without barriers, while encouraging registration by providing access to personalized features like creating listings and maintaining a wishlist.
By clearly delineating the actions available to new and registered customers, this diagram ensures a comprehensive understanding of user roles and their associated privileges. Additionally, the inclusion of external services
(ListingService and WishlistService) demonstrates how the platform integrates backend functionality to provide a seamless user experience.

4 System states

This section describes the various states of key business objects in the ReMarket platform. Each state represents a distinct stage in the lifecycle of an object, detailing the transitions triggered by specific events or actions. The state transitions are crucial for understanding how the system handles different processes and how each business object evolves from one stage to another.

4.1 Listing Object States

A listing (Figure 5) represents an item available for sale on the platform. The states of a listing include:

- Draft: The listing is being created and is not visible to other users. This is the initial state when a seller starts creating a listing. It allows the seller to input details such as title, description, price, and images without the listing being visible to potential buyers. The listing remains in this state until the seller is ready to publish it.
- Published: The listing is live and visible to potential buyers. Once the seller is satisfied with the listing details, they can publish it. At this stage, the item becomes available for browsing by all users of the platform, and buyers can express interest in purchasing it.
- Sold: The item has been purchased, and the listing is no longer active. Once the order is confirmed and the item is sold, the listing moves to this state. The listing will no longer be available for purchase by other users, and it may either be archived or deleted depending on the seller's preferences.

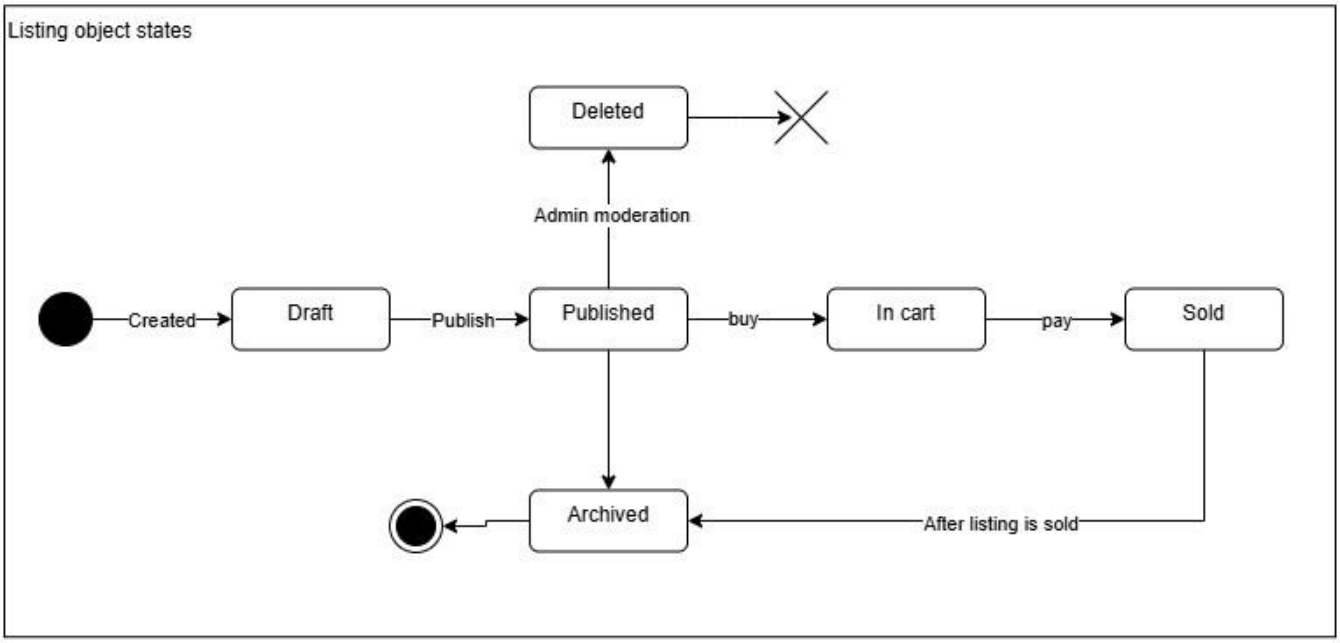


Figure 5: Listing object states diagram

- Archived: The listing is removed from public view but retained for record-keeping. This state is typically used for listings that are no longer available but should be kept for historical or administrative purposes. Archived listings are not visible to potential buyers but can still be accessed by the seller or platform administrators.
- Deleted: The listing is removed from public view but retained for record-keeping. It differs from *Archived* in a way that deleted listings cannot be viewed by the seller, in opposition to when it is in *Archived* state. This state is used when a listing is permanently removed from the platform, and the seller no longer has access to it.

4.2 Order Object States

An order (Figure 6) tracks the purchase process between a buyer and a seller. The states of an order include:

- Pending: The order has been created but is not yet confirmed. This is the initial state when a buyer places an order but has not yet received confirmation from the seller. The order is in a waiting state until the seller reviews and accepts it.
- Confirmed: The seller has accepted the order. After the seller reviews the order details and agrees to fulfill the purchase, the order transitions to this state. At this point, the seller is committed to shipping the item.
- Shipped: The item has been dispatched to the buyer. Once the seller has shipped the item, the order moves into the shipped state. This indicates that the item is on its way to the buyer.

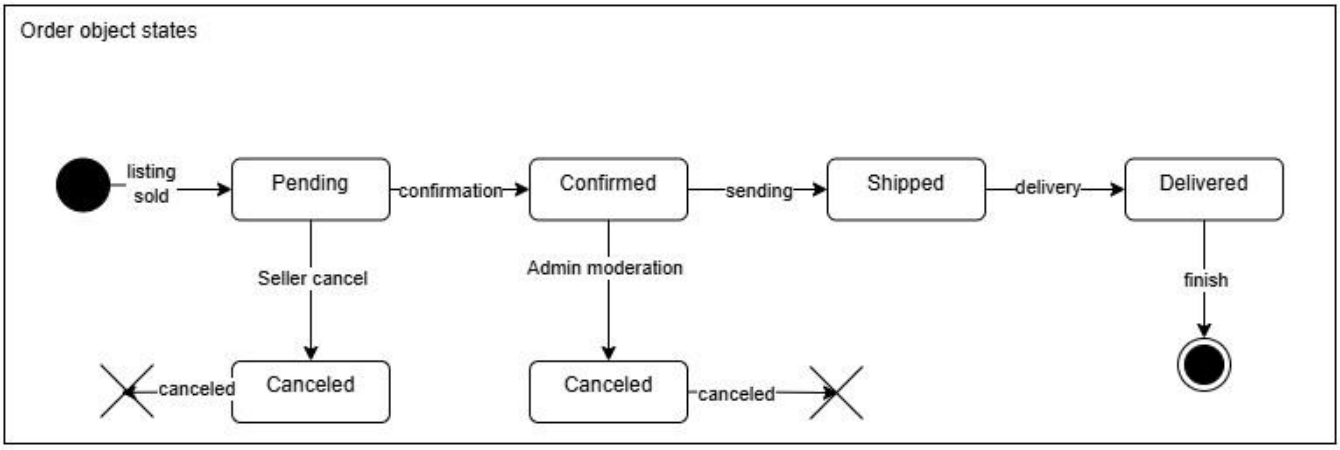


Figure 6: Order object states diagram

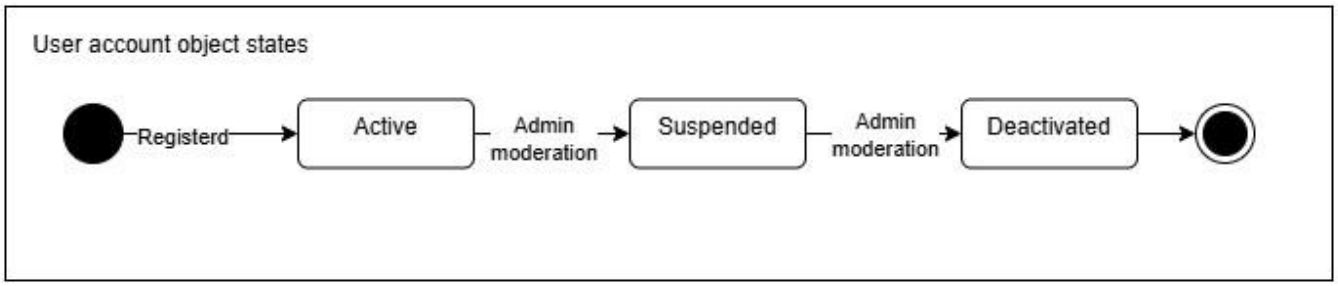


Figure 7: User account object states diagram

- Delivered: The buyer has received the item. When the item reaches the buyer and they confirm receipt, the order transitions to the delivered state. This is the final stage in the order lifecycle, indicating that the transaction has been completed successfully.
- Cancelled: The order has been cancelled by seller. If the buyer cancels the order before confirmation, or if the seller cancels due to availability issues or other reasons, the order enters this state. A cancelled order is no longer processed further.

4.3 User Account Object States

A user account (Figure 7) represents an individual's profile on the platform. The states of a user account include:

- Active: The account is in good standing and fully operational. This is the default state for most users. An active account allows the user to browse listings, make purchases, and interact with other users. It is the state in which the user can fully engage with the platform.

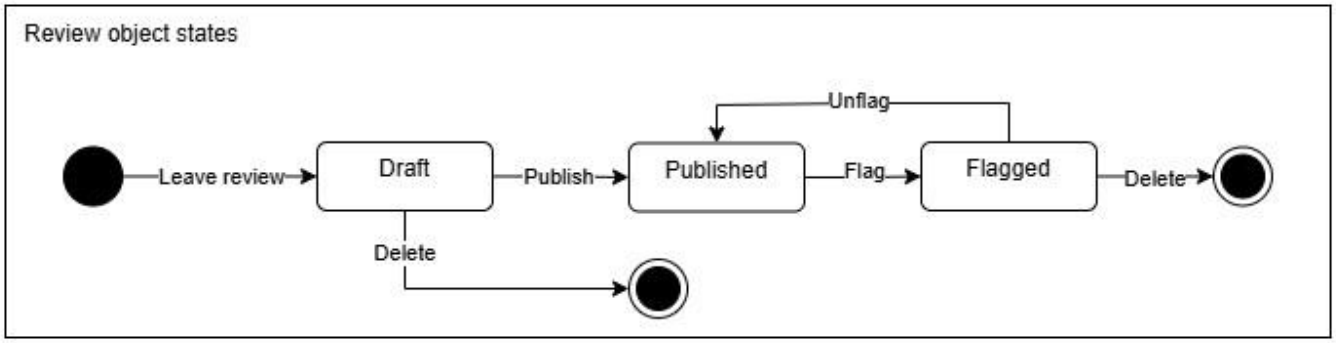


Figure 8: Review object states diagram

- Suspended: The account has been temporarily disabled due to policy violations. If a user violates platform policies, their account may be suspended. In this state, the user cannot perform certain actions, such as making purchases or selling items, until the suspension is lifted.
- Deactivated: The account has been permanently disabled due to policy violations. If a user repeatedly violates platform policies or engages in severe misconduct, their account may be permanently deactivated. In this state, the user can no longer access the platform or its services.

4.4 Review Object States

A review (Figure 8) represents feedback left by a user about a transaction. The states of a review include:

- Draft: The review is being written and is not yet submitted. Users can write and edit reviews before submitting them for public display. This state allows users to prepare their feedback and make changes before finalizing the review.
- Published: The review is visible to other users. Once the user submits the review, it becomes publicly visible to other users. This state allows others to see the feedback left by the reviewer, which may influence their purchasing decisions.
- Flagged: The review has been reported for inappropriate content. It can either be removed or be unflagged and become again available for public view. If a review is flagged by other users for violating platform guidelines (e.g., offensive language, spam, etc.), it enters this state. The platform administrators can then decide whether to remove or restore the review based on the investigation.

4.5 Category Object States

A category (Figure 9) helps organize listings into logical groups. The states of a category include:

- Active: The category is in use and visible to users. Categories are used to group similar listings together. When a category is active, it is visible on the platform and users can browse listings within that category.

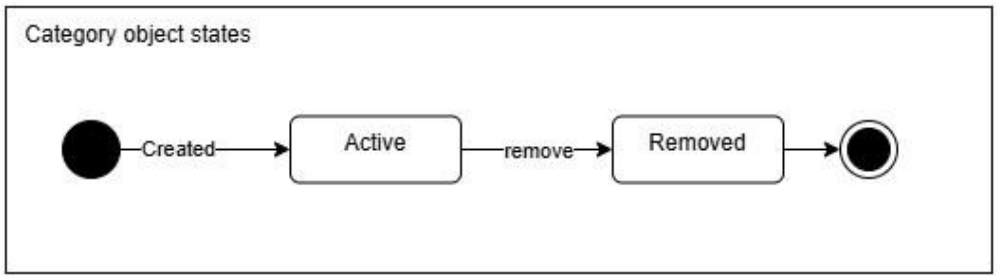


Figure 9: Category object states diagram

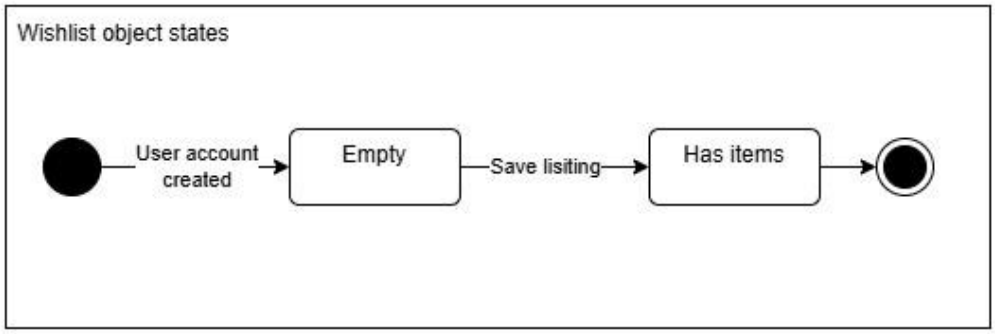


Figure 10: Wishlist object states diagram

- Removed: The category has been permanently removed. If a category is no longer relevant or is replaced by a new category, it may be removed. Once removed, the category is no longer available for browsing, and any listings within it may be reassigned to another category.

4.6 Wishlist Object States

A wishlist (Figure 10) represents a collection of items a user is interested in. The states of a wishlist include:

- Empty: wishlist is actively being used to track items. Users can add items to their wishlist and view them later. This state indicates that the wishlist is currently in empty.
- Has times: his state indicates that th The e wishlist is currently in non empty.

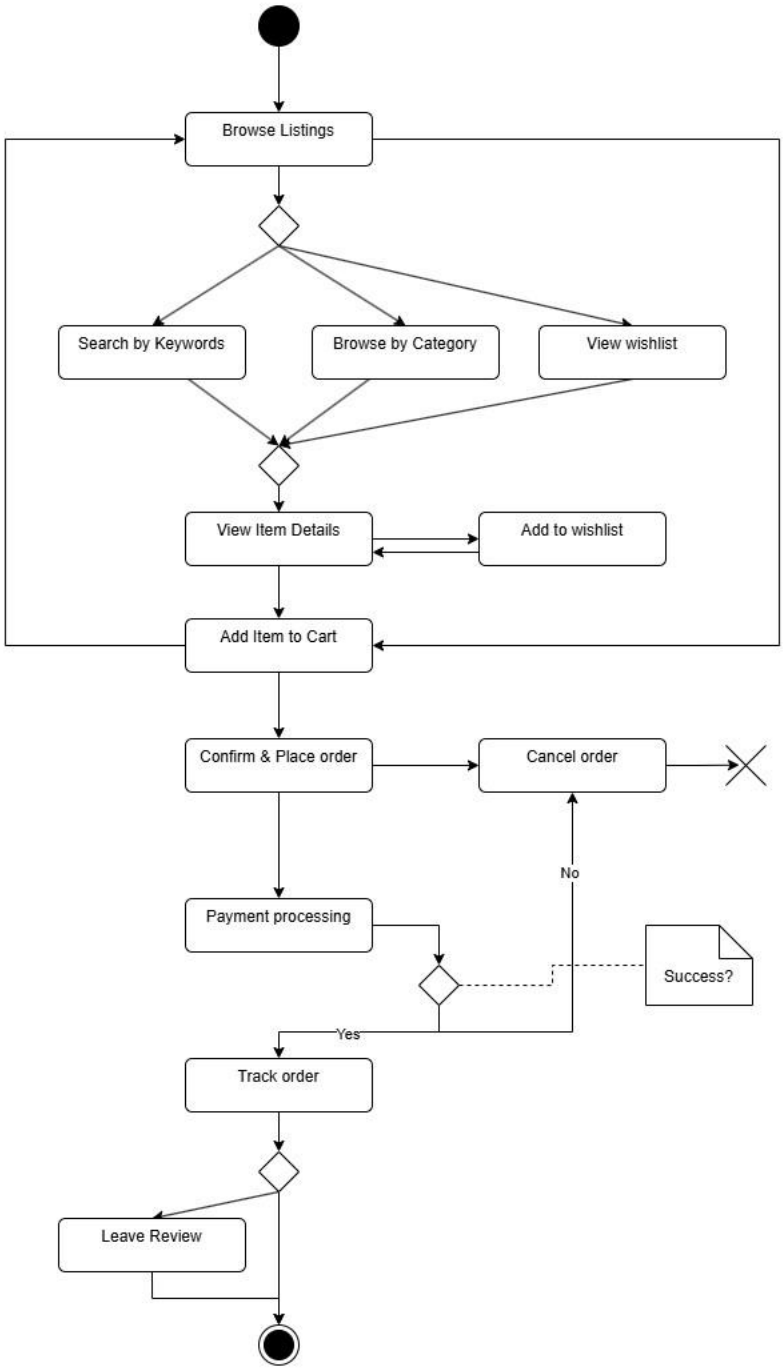


Figure 11: Buyer activity diagram

5 Activity Diagrams

5.1 Buyer activities

This section provides a detailed explanation of the buyer's activity flow in the ReMarket platform, as represented in the activity diagram (Figure 11). The diagram outlines the steps a buyer follows while browsing, selecting, and purchasing items, as well as managing orders and leaving reviews.

Overview of the Diagram

The activity diagram begins with the buyer entering the platform and browsing the available listings. It includes multiple decision points, actions, and flows that guide the user through the process of searching for items, adding items to the cart, placing an order, and tracking the purchase until completion. Additionally, the buyer has the option to manage orders, cancel purchases, and leave reviews. The flow is terminated when all activities are completed.

Activity Breakdown

The diagram can be broken down into the following key stages:

1. Browse Listings The process starts with the buyer entering the platform and initiating the action of browsing listings. This activity serves as the root action from which subsequent actions emerge. The buyer is presented with three options:

- Search by Keywords: The buyer searches for items using specific keywords.
- Browse by Category: The buyer navigates through predefined categories to find items of interest.
- View Wishlist: The buyer accesses their wishlist to review items they previously marked.

The flow converges after these actions, leading to the next stage where the buyer can view item details.

2. View Item Details and Add to Wishlist At this stage, the buyer can view the detailed information of a selected item. This includes the item's description, price, and other relevant attributes. From this point, the buyer has two choices:

- Add to Wishlist: The buyer can save the item for later consideration by adding it to their wishlist.
- Proceed to Add Item to Cart: The buyer decides to move forward with the purchase process by adding the item to their cart.

These actions are represented as bidirectional, indicating that the buyer can switch between viewing item details and adding the item to their wishlist.

3. Add Item to Cart Once the buyer decides to purchase an item, they add it to their cart. This action signifies the buyer's intent to proceed with the purchase process. At this point, the buyer can continue browsing or proceed to confirm and place the order.

4. Confirm and Place Order In this stage, the buyer reviews the items in their cart and confirms the purchase. Two key actions are possible here:

- Place Order: The buyer finalizes the order and initiates the payment process.
- Cancel Order: The buyer cancels the order, terminating the flow for that transaction.

If the order is placed, the flow moves to the payment processing stage.

5. Payment Processing After confirming the order, the payment process begins. A decision point determines whether the payment is successful:

- Success: If the payment is successful, the flow continues to the order tracking stage.
- Failure: If the payment fails, the buyer is presented with the option to retry the payment or cancel the order.

The decision is represented with a diamond node, showing the conditional flow based on the payment outcome.

6. Track Order Once the payment is successful, the buyer can track the order. This stage allows the buyer to monitor the status of their purchase, such as shipping and delivery updates. Tracking continues until the order is completed.

7. Leave Review After the order has been delivered, the buyer is presented with the option to leave a review. Leaving a review provides feedback about the transaction, which is visible to other users and sellers. This stage marks the final activity in the buyer's flow.

Flow Termination

The activity flow ends once the buyer leaves a review or decides not to take further action after tracking the order. The diagram uses a black circle to denote the termination of the activity.

Key Observations

The activity diagram highlights the following aspects of the buyer's interaction with the platform:

- The buyer's journey is flexible, allowing multiple entry points such as browsing listings, searching by keywords, or accessing the wishlist.
- The decision points, such as payment success or failure, ensure that all possible outcomes are accounted for, including retrying payments or canceling orders.
- The flow provides a logical progression from item discovery to purchase and post-purchase activities, ensuring a seamless user experience.
- Optional actions, such as adding items to the wishlist or leaving reviews, enhance user engagement without disrupting the primary purchase flow.

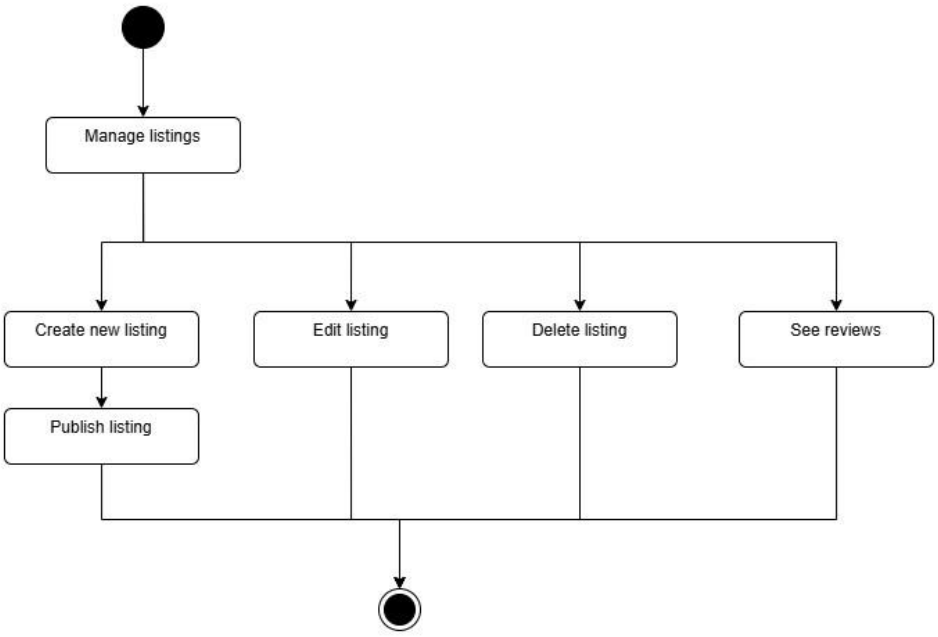


Figure 12: Seller activity diagram

5.2 Seller activities

The seller activity diagram (Figure 12) outlines the various actions available to sellers on the platform as part of their listing management process. Sellers play a crucial role in the marketplace by providing items for sale, maintaining the quality of listings, and interacting with buyers through reviews. The diagram captures the lifecycle of seller actions, beginning from managing their listings to performing operations such as creating, editing, deleting listings, and viewing buyer feedback.

Manage Listings. The activity flow begins with the seller accessing the Manage listings section. This is the central entry point for all seller-related operations on the platform. From here, sellers can perform multiple actions related to their listings, including creating new listings, modifying existing ones, removing outdated items, and reviewing feedback. The ability to manage listings efficiently ensures that sellers can maintain a well-organized and up-to-date inventory of items available for purchase.

Create New Listing. The Create new listing action allows sellers to add new items to the marketplace. This is the first step in making an item available for sale. During this process, sellers are prompted to provide essential details about the item, including:

- Title: A clear and concise name for the item.
- Description: A detailed explanation of the item's features, condition, and any other relevant information.
- Price: The cost of the item, which can be fixed or negotiable depending on the platform's rules.
- Images: High-quality images of the item to attract potential buyers.

The creation of a new listing is a critical activity, as the quality and accuracy of the listing details directly impact buyer interest. Once all required information has been entered, the seller can proceed to publish the listing.

Publish Listing. The Publish listing action marks the transition of a listing from a draft state to being publicly visible. Once published, the listing becomes available to potential buyers who can view its details, express interest, or proceed with a purchase. Publishing ensures that the item is live on the platform and accessible to all users. Sellers may choose to delay publishing if they need to refine details or gather additional information about the item.

Edit Listing. The Edit listing action provides sellers with the flexibility to update existing listings. Editing allows sellers to modify any aspect of the listing, such as:

- Adjusting the price to respond to market conditions or buyer feedback.
- Updating the description to provide clearer or more accurate information.
- Replacing or adding images to better showcase the item's condition.

This action is particularly useful when sellers receive feedback or notice that the listing requires improvements to attract more buyers. Keeping listings updated helps maintain trust and transparency within the platform.

Delete Listing. If a seller decides that a listing is no longer needed, they can perform the Delete listing action. Deleting a listing removes it from public view and ensures that it is no longer available for buyers to interact with.
Sellers may choose to delete a listing for several reasons, including:

- The item has been sold outside the platform.
- The item is no longer available for sale.
- The listing contains incorrect or outdated information that cannot be easily corrected.

Deleting a listing helps sellers maintain a clean and accurate inventory, ensuring that only active and relevant listings are visible to buyers.

See Reviews. The See reviews action allows sellers to view feedback provided by buyers. Reviews play a significant role in the marketplace, as they reflect the buyer's experience with the seller and the quality of the purchased item. Sellers can use reviews to:

- Identify areas for improvement, such as more accurate item descriptions.
- Build trust and credibility with potential buyers by maintaining a positive review history.
- Address any concerns raised by buyers to enhance their overall selling process.

By regularly reviewing feedback, sellers can continuously improve their listings and ensure a positive experience for future buyers.

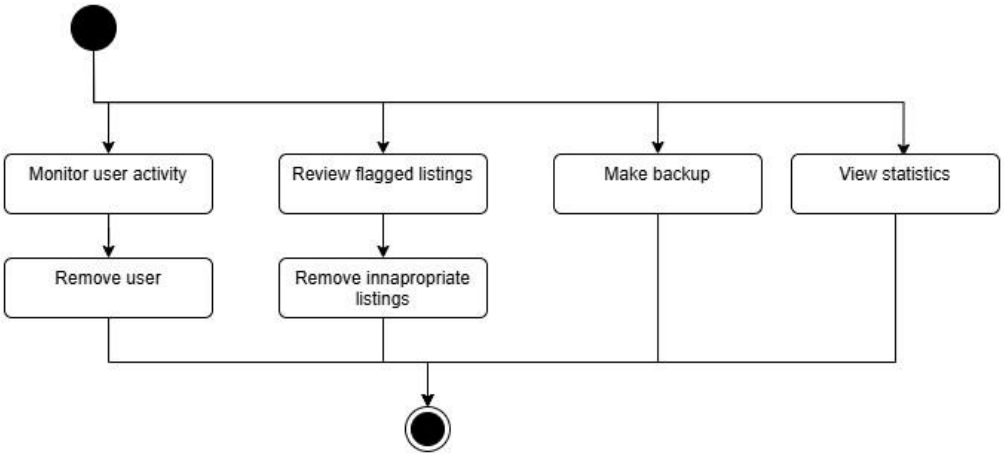


Figure 13: Admin activity diagram

Key Observations

The seller activity diagram (Figure 12) highlights several key observations regarding the actions and workflows available to sellers on the platform:

- Centralized Management: All seller actions originate from the *Manage listings* entry point, providing a single interface for performing various operations. This centralization simplifies navigation and ensures sellers can efficiently manage their listings.
- Listing Lifecycle: The diagram captures the lifecycle of a listing, from creation to publication and eventual deletion. This flow emphasizes the flexibility sellers have in managing their items, including the ability to update details or remove outdated listings.
- Separation of Actions: Each seller action—*Create new listing*, *Edit listing*, *Delete listing*, and *See reviews*—is clearly separated. This modularity allows sellers to focus on specific tasks without unnecessary complexity.
- Publishing as a Key Transition: The transition from *Create new listing* to *Publish listing* is significant, as it moves a listing from a draft state to being visible to buyers. Sellers have the flexibility to delay publication until they are satisfied with the listing details.
- Importance of Reviews: The *See reviews* action highlights the role of buyer feedback in the seller's workflow. Reviews provide valuable insights for sellers, helping them improve their performance, build trust, and maintain a positive reputation.

5.3 Admin activities

The administrator activity diagram (Figure 13) outlines the key actions performed by administrators on the platform to maintain the integrity, functionality, and overall quality of the marketplace. Administrators play a vital role in ensuring compliance with platform rules, monitoring user behavior, and safeguarding the system's operational stability. The diagram captures the sequence of administrative activities, beginning with monitoring users and listings to performing essential maintenance and statistical review.

Monitor User Activity. The workflow begins with the administrator accessing the Monitor user activity functionality. This is a critical task where administrators oversee user behavior, focusing on actions such as item listings, interactions, transactions, and potential policy violations. Monitoring user activity allows administrators to identify and address suspicious activities that could harm the platform's reputation or user experience.

Remove User. The Remove user action is a consequence of monitoring user activity. If an administrator finds evidence of misconduct, such as fraudulent activity or repeated violations of the platform's policies, they can take action to remove the offending user from the platform. This ensures that the marketplace remains safe and trustworthy for all participants.

Review Flagged Listings. Another significant activity is the Review flagged listings process. Listings flagged by users or automatically identified by the system for potential violations (e.g., inappropriate content, false information, or prohibited items) are reviewed by the administrator. This step ensures that flagged content receives attention and that decisions are made in accordance with the platform's guidelines.

Remove Inappropriate Listings. The Remove inappropriate listings action allows administrators to delete flagged listings deemed unsuitable for the platform. Removing such listings prevents harm to the platform's reputation and ensures compliance with marketplace policies. It also maintains the quality of available items, fostering a better experience for buyers and sellers.

Make Backup. The Make backup action focuses on system maintenance and data protection. Administrators can create backups of critical data, including user information, listings, transactions, and activity logs. Backups ensure data recovery in case of unexpected system failures, enhancing the platform's reliability and resilience.

View Statistics. The View statistics functionality provides administrators with insights into the platform's performance and user engagement. By analyzing metrics such as user activity, listing trends, and overall platform usage, administrators can make data-driven decisions to optimize the platform's operations and identify areas for improvement.

6 System Communication

Effective system communication is integral to ensuring seamless interactions between various components of a software system. This section provides an in-depth analysis of the communication mechanisms utilized in the project, focusing on the interactions facilitated by the REST API. The REST API serves as the backbone of the system, enabling modular, scalable, and efficient communication between clients and the server.

To illustrate this, sequence diagrams have been employed to visualize the flow of interactions for key functionalities, complemented by a detailed overview of the API endpoints responsible for these operations. The following subsections analyze each diagram and its associated API routes, highlighting the design principles and implementation details underpinning the communication logic.

6.1 Authentication Scenarios

The authentication process is a critical component of the system, ensuring secure access and registration for users. The sequence diagram in Figure 14 visualizes the flow of communication between the Web Customer, Authentication Service, and the Database during login and registration scenarios. The REST API endpoints responsible for these actions are detailed below.

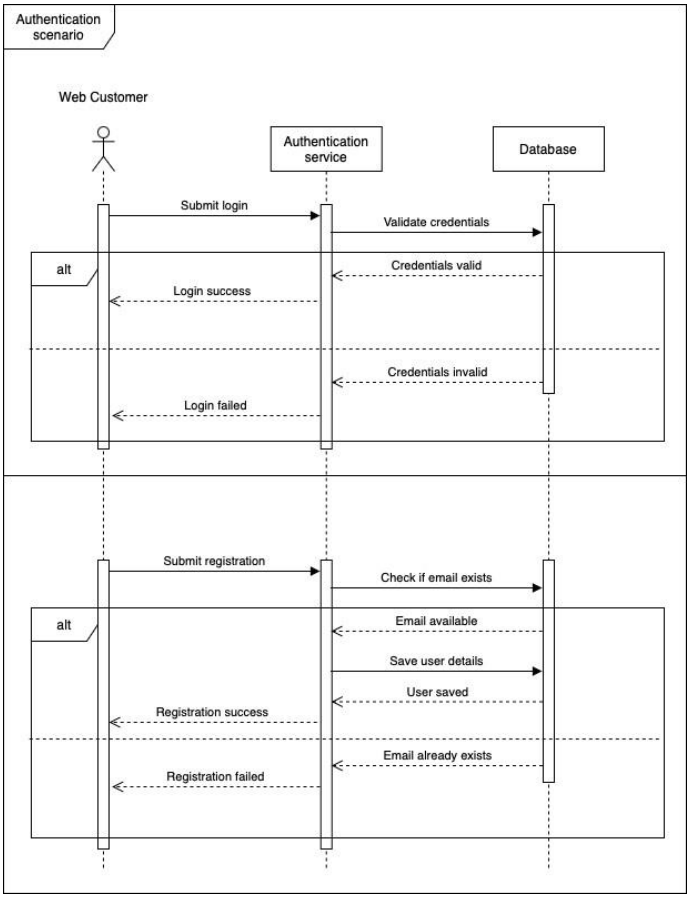


Figure 14: Authentication Scenario - Login and Registration

6.1.1 Login Process

The login process begins when a user submits their credentials via the POST /auth endpoint. The Authentication Service validates the credentials against the Database. Based on the validation results:

- Login Success: If the credentials are valid, the service returns a success response with a token (HTTP 200). Example response:

```
{
   "token": "abcdef123456",
   "expiresIn": 3600
}
```

- Login Failure: If the credentials are invalid, the service returns an unauthorized error (HTTP 401). Example response:

```
{
   "message": "Invalid email or password." }
```

6.1.2 Registration Process

The registration process allows a new user to create an account using the POST /auth/register endpoint. The sequence diagram highlights two outcomes:

- Registration Success: If the provided email is not already in use, the service saves the user details to the Database and returns a success response (HTTP 201). Example response:

```
{
   "userId": "user123",
   "token": "abcdef123456"
}
```

- Registration Failure: If the email already exists, the service returns a bad request error (HTTP 400). Example response:

```
{
   "message": "Email already exists." }
```

6.1.3 Reset Password Process

The system also supports password recovery via the POST /auth/reset-password endpoint. When a user provides their email address:

- Success: If the email is registered, a password reset link is sent, and a success response is returned (HTTP 200). Example response:

```
{
   "message": "Password reset link sent to your email." }
```

- Failure: If the email is not found in the system, an error response is returned (HTTP 404). Example response:

```
{
   "message": "Email not found."
}
```

6.2 Browsing Scenario

The browsing process enables users to explore product listings, apply filters, and add items to their wishlist or shopping cart. Figure 15 illustrates the sequence of interactions involving the Web Customer, vicevice, Wishlist Service, and Wishlist Service.

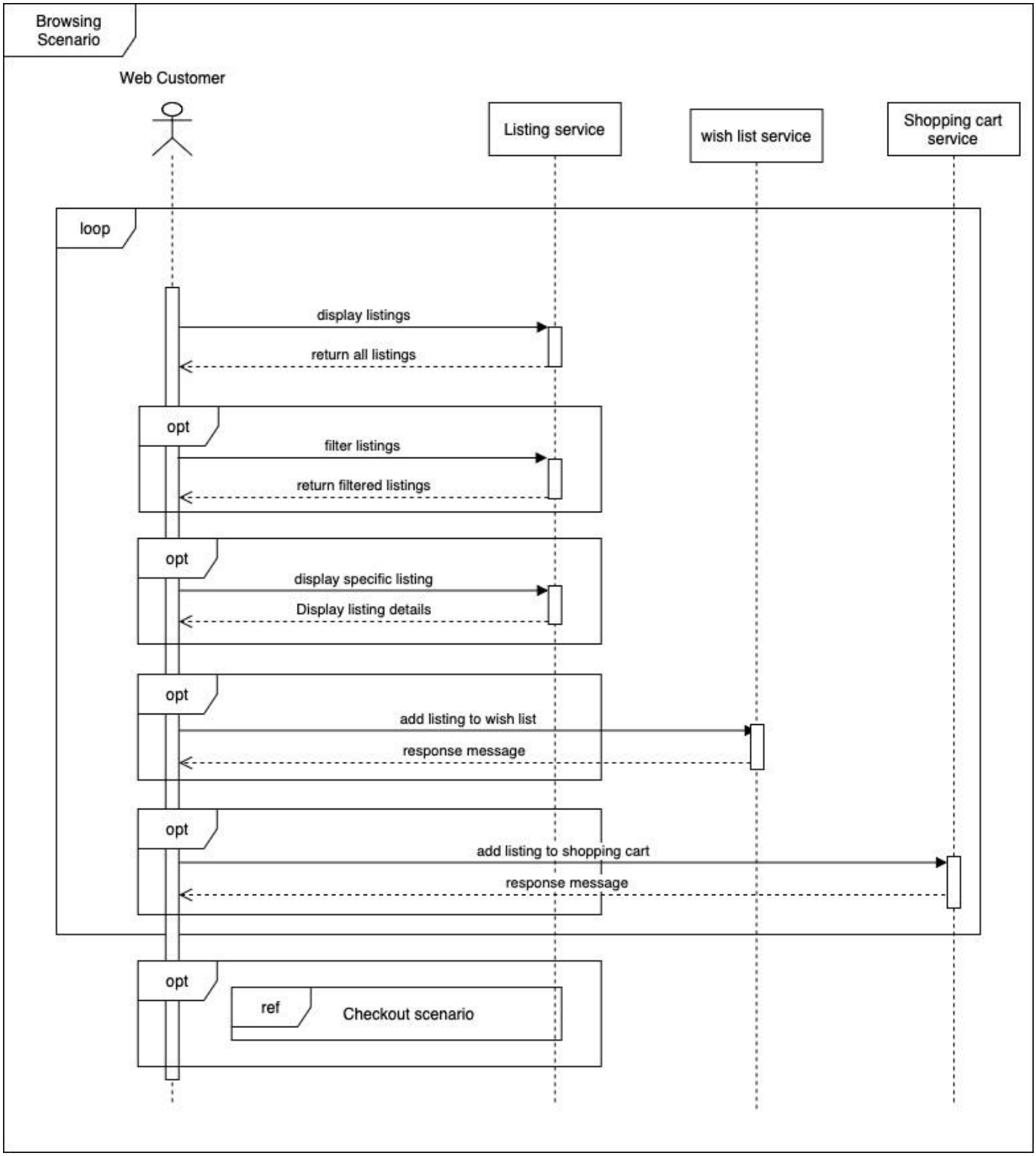


Figure 15: Browsing Scenario - Viewing and Managing Listings

6.2.1 Retrieve Listings

Users can retrieve listings using the GET /listings endpoint. Optional query parameters allow users to filter by name, price range, category, and paginate through results. A successful request (HTTP 200) returns a list of listings. Example response:


```
{
   "listings": [
      {
         "id": "listing789",
         "name": "Vintage Camera",
         "price": 150.00,
         "description": "A classic film camera from the 1960s.",
         "category": "Electronics",
         "images": [
            "https://api.remarket.com/images/listings/listing789_1.jpg",
            "https://api.remarket.com/images/listings/listing789_2.jpg" ]
      }
   ],
   "page": 1,
   "pageSize": 10,
   "total": 1
}
```


6.2.2 View Listing Details

Specific listing details can be retrieved using the GET /listings/{listingId} endpoint. This operation returns the full details of the listing, including its name, price, description, and associated images. Example response:


```
{
   "id": "listing789",
   "name": "Vintage Camera",
   "price": 150.00,
   "description": "A classic film camera from the 1960s.",
   "category": "Electronics",
   "images": [
      "https://api.remarket.com/images/listings/listing789_1.jpg",
      "https://api.remarket.com/images/listings/listing789_2.jpg" ]
}
```


If the listing ID is invalid or does not exist, an error response (HTTP 404) is returned.

6.2.3 Add to Wishlist

Listings can be added to a user's wishlist using the POST /listing/{listingId}/wishlist endpoint. A success response (HTTP 201) confirms the addition. Example response:


```
{
   "message": "Item added to wishlist."
}
```

If the listing does not exist, the service returns an error (HTTP 404). If the item is already in the wishlist, a bad request response (HTTP 400) is returned.

6.2.4 Add to Shopping Cart

Similarly, users can add listings to their shopping cart using the POST /listing/{listingId}/shopping-cart endpoint. The request body includes the quantity of the item. A successful request (HTTP 201) confirms the operation. Example response:


```
{
   "message": "Item added to shopping cart."
}
```


Invalid quantity or a nonexistent listing results in error responses (HTTP 400 or HTTP 404).

6.2.5 Filtered and Paginated Listings

The GET /listings endpoint supports additional filters such as minimum and maximum price, category, and name. Pagination parameters include page and pageSize, allowing users to manage large datasets efficiently.

GET /listings?minPrice=50&maxPrice=200&category=Electronics&page=1&pageSize=10 Example response:


```
{
   "listings": [...],
   "page": 1,
   "pageSize": 10,
   "total": 50
}
```


6.3 Wishlist Scenario

The wishlist scenario enables users to view items saved in their wishlist, manage these items, and interact with them, such as removing items or proceeding to the checkout process. Figure 16 illustrates the sequence of operations between the Web Customer and the Wishlist Service.

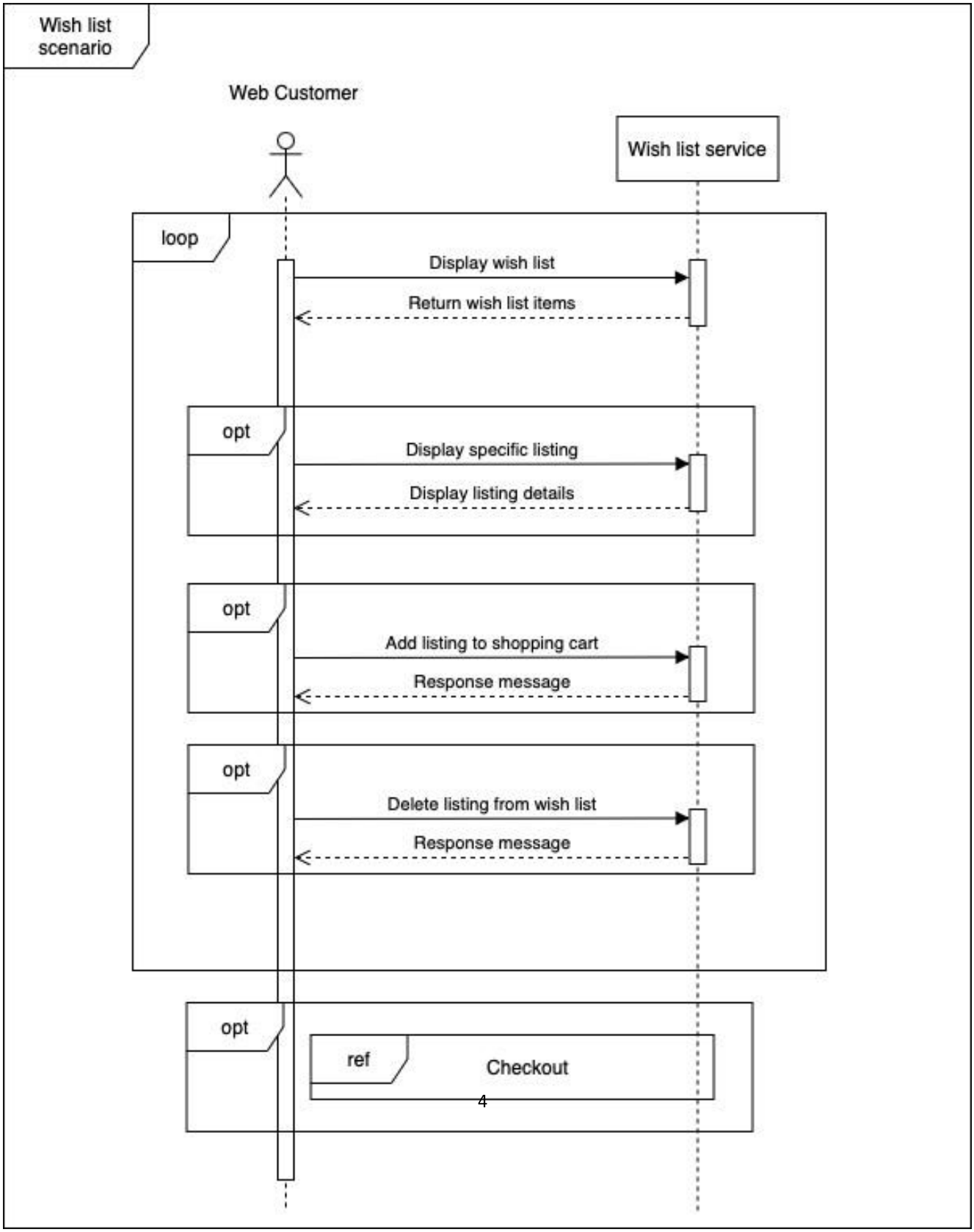

Wish list scenario
Web Customer
Wish list service
loop
Display wish list
Return wish list items
opt
Display specific listing
Display listing details
opt
Add listing to shopping cart
Response message
opt
Delete listing from wish list
Response message
opt
ref
Checkout

Figure 16: Wishlist Scenario - Viewing and Managing Wishlist Items

6.3.1 Retrieve Wishlist Items

Users can retrieve all items stored in their wishlist using the GET /wishlist endpoint. A successful response (HTTP 200) returns a list of wishlist items. Example response:


```
{
  "message": "Wishlist retrieved successfully.",
  "items": [
    {
      "listingId": "listing789",
      "name": "Vintage Camera",
      "price": 150.00
    },
    {
      "listingId": "listing790",
      "name": "Handmade Necklace",
      "price": 45.00
    }
  ]
}
```


If the user is not authenticated, the service returns an unauthorized access error (HTTP 401). Example response:


```
{
  "message": "Authentication required."
}
```


6.3.2 Remove Item from Wishlist

Users can remove an item from their wishlist using the DELETE /wishlist/{listingId} endpoint. The listingId parameter identifies the specific item to be deleted. A successful deletion operation returns no content (HTTP 204). If the item does not exist or is invalid, the service returns an error response (HTTP 404). Example response:


```
{
  "message": "Item not found."
}
```


If the user is not authenticated, the service responds with an unauthorized access error (HTTP 401).

6.3.3 Checkout Process

While the wishlist does not directly handle the checkout process, the diagram references the "Checkout Scenario" as a next step in the flow. Items added to the shopping cart during the wishlist interactions can proceed to the checkout process via the Shopping Cart Service, described in the shopping cart scenario.

6.4 Checkout Scenario

The checkout scenario is the final step in the user journey, where items in the shopping cart are processed into an order. This process involves interactions with the Checkout Service, Payment Gateway, and Shipping Company, as illustrated in Figure 17.

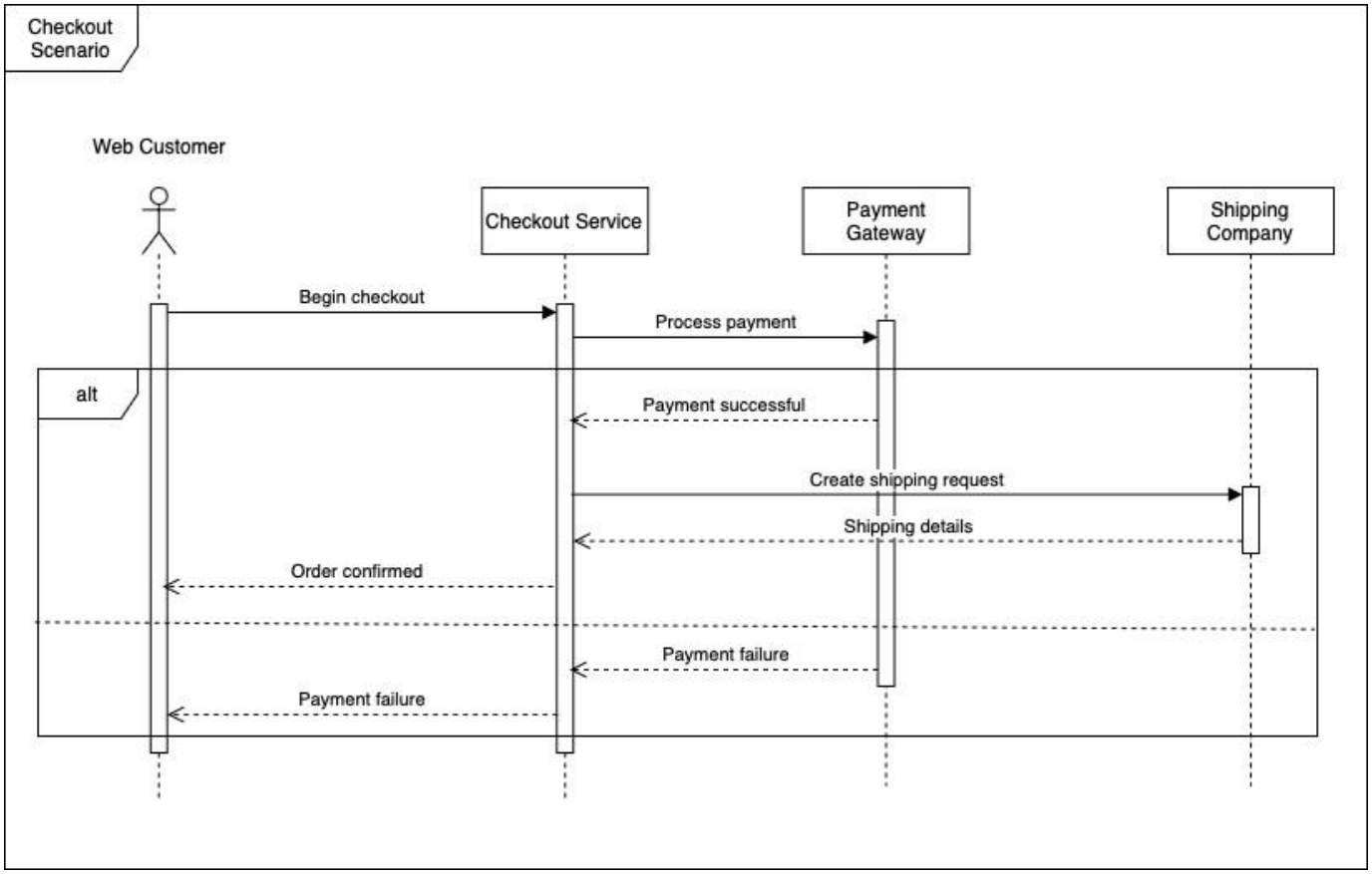


Figure 17: Checkout Scenario - Order Placement and Fulfillment

6.4.1 Retrieve Shopping Cart Items

Users can view the contents of their shopping cart using the GET /shopping-cart endpoint. A successful response (HTTP 200) includes a list of items in the cart. Example response:


```
{
  "message": "Shopping cart retrieved successfully.",
  "items": [
    {
      "listingId": "listing789",
      "name": "Vintage Camera",
      "price": 150.00,
      "quantity": 1
    },
    {
```

```
            "listingId": "listing790",
            "name": "Handmade Necklace",
            "price": 45.00,
            "quantity": 2
        }
    ]
}
```


If the user is not authenticated, the service returns an unauthorized error (HTTP 401). Example response:


```
{
    "message": "Authentication required."
}
```


6.4.2 Begin Checkout Process

To create an order from the items in the shopping cart, the user interacts with the POST /shopping-cart/checkout endpoint. The request payload includes shipping details, such as the address, city, postal code, and country as well as the choosen shipping method. A successful request (HTTP 201) returns the order details, including the order ID, status, and total amount. Example response:


```
{
    "orderId": "order001",
    "status": "Processing",
    "totalAmount": 240.00
}
```


If the provided checkout details are invalid, the service responds with a bad request error (HTTP 400). Example response:


```
{
    "message": "Invalid checkout details."
}
```


6.4.3 Payment Processing

As shown in Figure 17, the Checkout Service processes the payment by interacting with the Payment Gateway. If the payment is successful, the system confirms the order and proceeds to create a shipping request with the Shipping Company. The user receives a confirmation message indicating that the order has been placed. In the event of a payment failure, the Checkout Service notifies the user, and the process terminates with an error.

6.4.4 Shipping Request

Once the payment is confirmed, the Checkout Service communicates with the Shipping Company to create a shipping request. The shipping details, such as tracking information, are then returned to the Checkout Service, completing the order fulfillment process.

6.5 Tracking Order Scenario

The order tracking scenario allows users to monitor the status of their orders and leave reviews for completed transactions. Figure 18 illustrates the interaction between the Web Customer, Order Tracking Service, and Review Service.

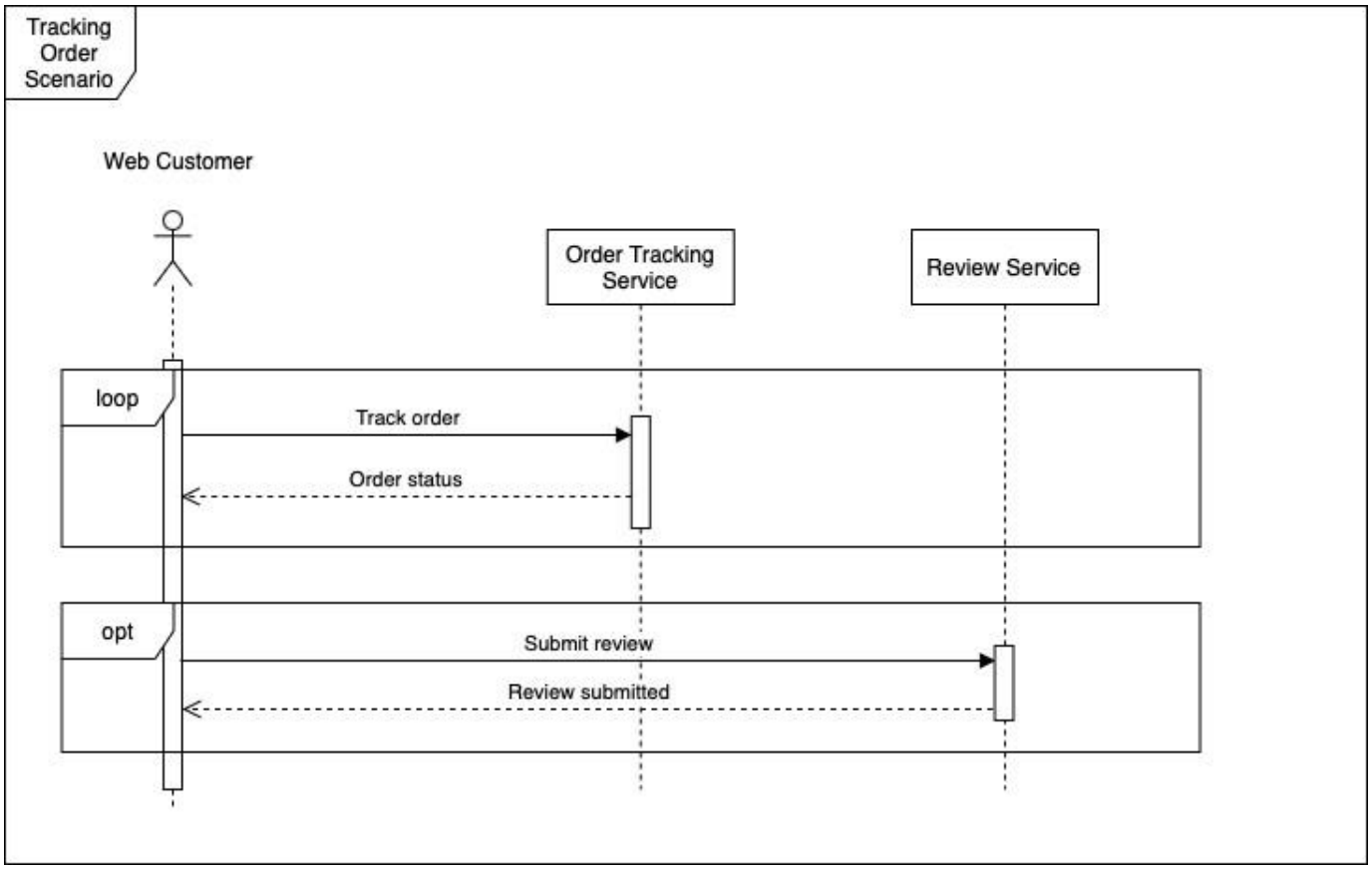


Figure 18: Tracking Order Scenario - Order Monitoring and Review Submission

6.5.1 Retrieve Orders

Users can retrieve a list of their orders using the GET /orders endpoint. A successful response (HTTP 200) returns a paginated list of orders with their current status. Example response:

```
{
   "orders": [
      {
         "orderId": "order001",
         "status": "Processing",
         "totalAmount": 240.00
```

```
        }
    ],
    "page": 1,
    "pageSize": 10,
    "total": 1
}
```


If the user is not authenticated, the service returns an unauthorized error (HTTP 401). Example response:


```
{
    "message": "Authentication required."
}
```


6.5.2 Retrieve Order Details

Specific order details can be retrieved using the GET /orders/{orderId} endpoint. This includes information about the items in the order, their quantities, and the total amount. A successful response (HTTP 200) includes detailed order information. Example response:


```
{
    "orderId": "order001",
    "items": [
        {
            "listingId": "listing789",
            "name": "Vintage Camera",
            "price": 150.00,
            "quantity": 1
        },
        {
            "listingId": "listing790",
            "name": "Handmade Necklace",
            "price": 45.00,
            "quantity": 2
        }
    ],
    "totalAmount": 240.00,
    "status": "Processing"
}
```


If the order does not exist, the service returns a 404 error. If the user is unauthorized or forbidden from viewing the order, appropriate 401 or 403 errors are returned.

6.5.3 Submit Review

Once an order is completed, the user can leave a review for the seller using the POST /orders/{orderId}/review endpoint. The request includes a rating and an optional comment. A successful submission (HTTP 201) confirms the review. Example response:

{
"message": "Review submitted successfully."
}

If the request contains invalid data, such as an out-of-range rating, the service returns a bad request error (HTTP 400). If the user is unauthorized or forbidden from reviewing the order, 401 or 403 errors are returned. If the order does not exist, a 404 error is returned. Example error response:

{
"message": "Order not found."
}

6.5.4 Track Order Status

Users can track the real-time status of their orders through the sequence diagram depicted in Figure 18. The loop interaction shows repeated requests to the Order Tracking Service for updates. Responses indicate whether the order is in processing, shipped, or delivered.

6.6 Seller Scenario

The seller scenario focuses on managing product listings, allowing sellers to create, edit, and delete their listings. Figure 19 illustrates the interactions between the Seller and the Listing Service for managing listings.

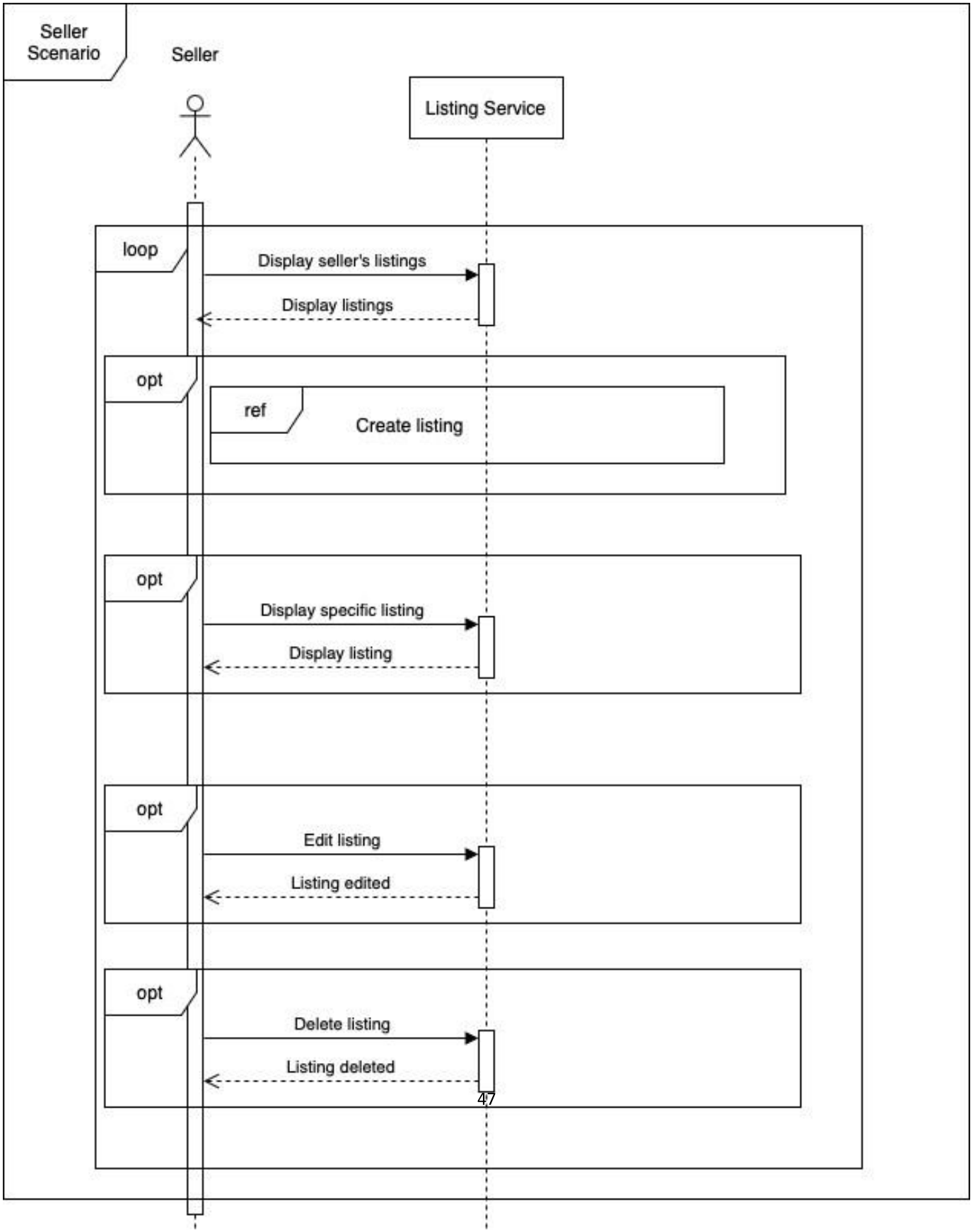

Seller Scenario
Seller
Listing Service
loop
Display seller's listings
Display listings
opt
ref
Create listing
opt
Display specific listing
Display listing
opt
Edit listing
Listing edited
opt
Delete listing
Listing deleted

Figure 19: Seller Scenario - Managing Listings

6.6.1 Retrieve Seller's Listings

Sellers can view all listings they have created using the GET /users/{userId}/listings endpoint. This endpoint returns a paginated list of their listings. Example response:


```
{
   "listings": [
      {
         "id": "listing789",
         "name": "Vintage Camera",
         "price": 150.00,
         "description": "A classic film camera from the 1960s.",
         "category": "Electronics",
         "images": [
            "https://api.remarket.com/images/listings/listing789_1.jpg",
            "https://api.remarket.com/images/listings/listing789_2.jpg" ]
      }
   ],
   "page": 1,
   "pageSize": 10,
   "total": 1
}
```


If the seller is not authenticated, the service returns an unauthorized error (HTTP 401). If the user ID does not exist, a 404 error is returned.

6.6.2 Create Listing

Sellers can add new listings using the POST /my-listings endpoint. The request payload includes details such as the name, price, description, category, and images of the product. A successful request (HTTP 201) returns the details of the newly created listing. Example response:


```
{
   "id": "listing791",
   "name": "Vintage Radio",
   "price": 80.00,
   "description": "An old radio from the 1950s.",
   "category": "Electronics",
   "images": [
      "https://api.remarket.com/images/listings/listing791_1.jpg" ]
}
```


If required fields are missing or invalid, the service returns a 400 error. Example response:


```
{
   "message": "Missing required listing fields."
}
```


6.6.3 Edit Listing

To modify an existing listing, sellers use the PUT /listings/{listingId} endpoint. The request payload can include updated values for the listing name, price, description, category, or images. A successful request (HTTP 200) returns the updated listing details. Example response:

```
{
  "id": "listing789",
  "name": "Vintage Camera (Updated)",
  "price": 140.00,
  "description": "A classic film camera from the 1960s, in excellent condition.",
  "category": "Electronics",
  "images": [
    "https://api.remarket.com/images/listings/listing789_1_updated.jpg"
  ]
}
```


If the listing ID is invalid or not found, the service returns a 404 error. If the seller is not authorized to edit the listing, a 403 error is returned.

6.6.4 Delete Listing

Sellers can delete a listing they created using the DELETE /listings/{listingId} endpoint. A successful deletion returns a 204 status with no content.

If the listing does not exist or the seller is not authorized to delete it, appropriate 404 or 403 errors are returned.

6.7 Create Listing Scenario

The create listing scenario allows sellers to add new items to their catalog. The process involves submitting item details, uploading photos, setting prices, and configuring shipping options. Figure 20 illustrates the sequence of interactions between the Seller and the Listing Service during this process.

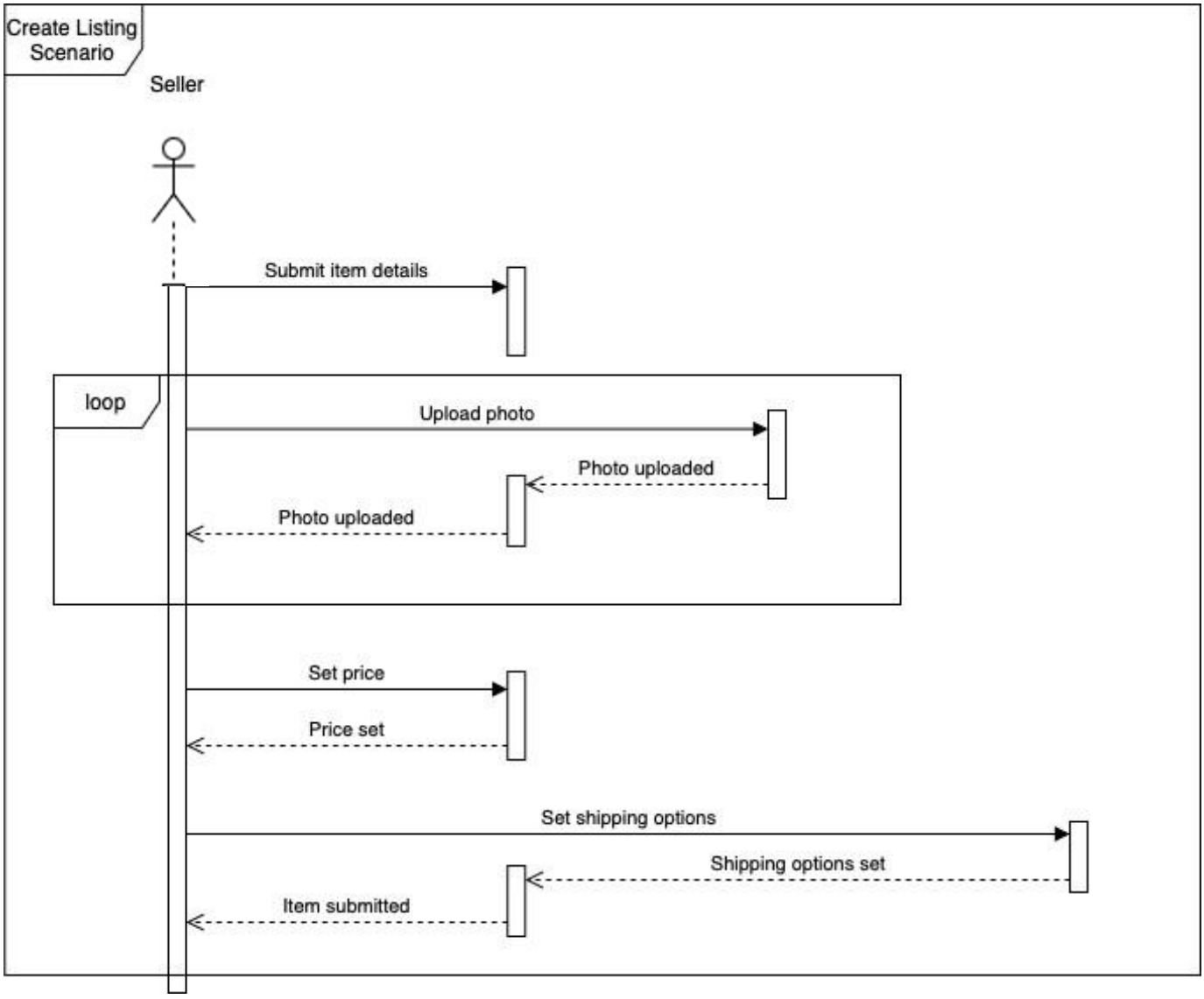


Figure 20: Create Listing Scenario - Adding a New Listing

6.7.1 Submit Item Details

The process begins with the seller submitting basic item details, such as the name, category, and description, through the POST /my-listings endpoint. This initial submission also includes one or more images of the item. A successful response (HTTP 201) returns the details of the newly created listing. Example response:


```
{
   "id": "listing791",
   "name": "Vintage Radio",
   "price": 80.00,
   "description": "An old radio from the 1950s.",
   "category": "Electronics",
   "images": [
      "https://api.remarket.com/images/listings/listing791_1.jpg" ]
}
```

If any required fields are missing, the service responds with a 400 Bad Request. Example error response:

```
{
  "message": "Missing required listing fields."
}
```

6.7.2 Upload Photos

Sellers can upload multiple photos for the listing as part of the item creation process. Photos are submitted as part of the multipart form-data request to the POST /my-listings endpoint. Each uploaded photo is stored and linked to the listing. A successful upload ensures the listing is visually descriptive for potential buyers.

6.7.3 Set Price

After submitting the basic details, the seller sets a price for the item. This is a required field in the request payload, and the service validates the price to ensure it is within an acceptable range. Any invalid price results in a 400 Bad Request response.

6.7.4 Set Shipping Options

Finally, the seller configures shipping options for the listing, including the shipping methods available and estimated delivery times. These options are also included in the submission via the POST /my-listings endpoint.

6.7.5 Complete Listing Creation

Once all fields are submitted and validated, the item is successfully added to the seller's catalog, and the process concludes. This is depicted in the sequence diagram as the final "Item submitted" interaction.

6.8 Update Profile Scenario

The update profile scenario allows users to modify their personal details, such as updating their bio and uploading a new profile picture. Figure 21 illustrates the interactions between the User, Profile Service, and Storage Service during this process.

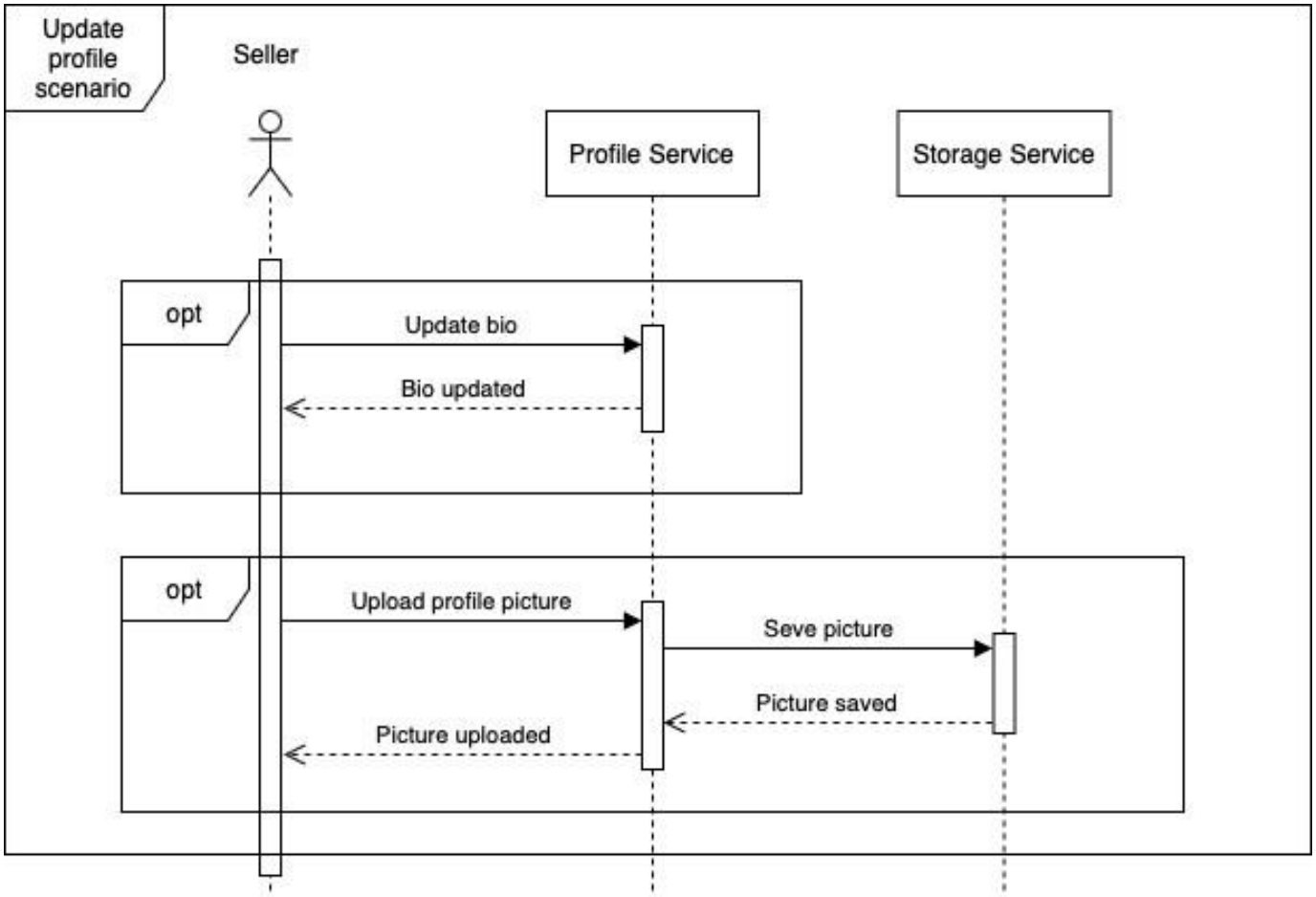


Figure 21: Update Profile Scenario - Modifying User Profile

6.8.1 Retrieve User Profile

Users can view their current profile details using the GET /users/me endpoint. This request retrieves information such as the user's bio, profile picture, and account status. A successful response (HTTP 200) returns the user profile. Example response:

```
{
  "userId": "user123",
  "name": "John Doe",
  "email": "johndoe@example.com",
  "bio": "I love selling vintage items.",
  "picture": "https://api.remarket.com/images/users/user123.jpg",
  "isSuspended": false,
  "suspendedUntil": null,
  "isBanned": false
}
```

If the user is not authenticated, an HTTP 401 error is returned. Example error response:

```
{
  "message": "Authentication required."
}
```

6.8.2 Update Bio

Users can update their bio using the PUT /users/me endpoint. The bio is submitted as part of a multipart form-data request. A successful update (HTTP 200) returns the updated profile. Example response:


```
{
  "userId": "user123",
  "name": "John Doe",
  "email": "johndoe@example.com",
  "bio": "Updated bio",
  "picture": "https://api.remarket.com/images/users/user123.jpg",
  "isSuspended": false,
  "suspendedUntil": null,
  "isBanned": false
}
```


If the bio format is invalid, a 400 Bad Request error is returned. Example error response:


```
{
  "message": "Invalid bio format."
}
```


6.8.3 Upload Profile Picture

Users can upload or update their profile picture via the same PUT /users/me endpoint. The picture is submitted as a file within the multipart form-data request. The Profile Service interacts with the Storage Service to save the image, and a successful request (HTTP 200) returns the updated profile with the new picture. Example response:


```
{
  "userId": "user123",
  "name": "John Doe",
  "email": "johndoe@example.com",
  "bio": "Updated bio",
  "picture": "https://api.remarket.com/images/users/user123_updated.jpg",
  "isSuspended": false,
  "suspendedUntil": null,
  "isBanned": false }
```


If the file format is invalid or corrupted, a 400 Bad Request error is returned. Example error response:


```
{
  "message": "Invalid picture format."
}
```


6.8.4 Complete Profile Update

Once the bio and/or profile picture are successfully updated, the user profile reflects the changes, completing the process. The sequence diagram highlights these steps, showing the interactions between the Profile Service and the Storage Service for handling profile data.

6.9 Manage Users Scenario

The manage users scenario allows administrators to view and manage user accounts, including suspending or banning users. Figure 22 illustrates the interactions between the Admin and the Account Service for managing users.

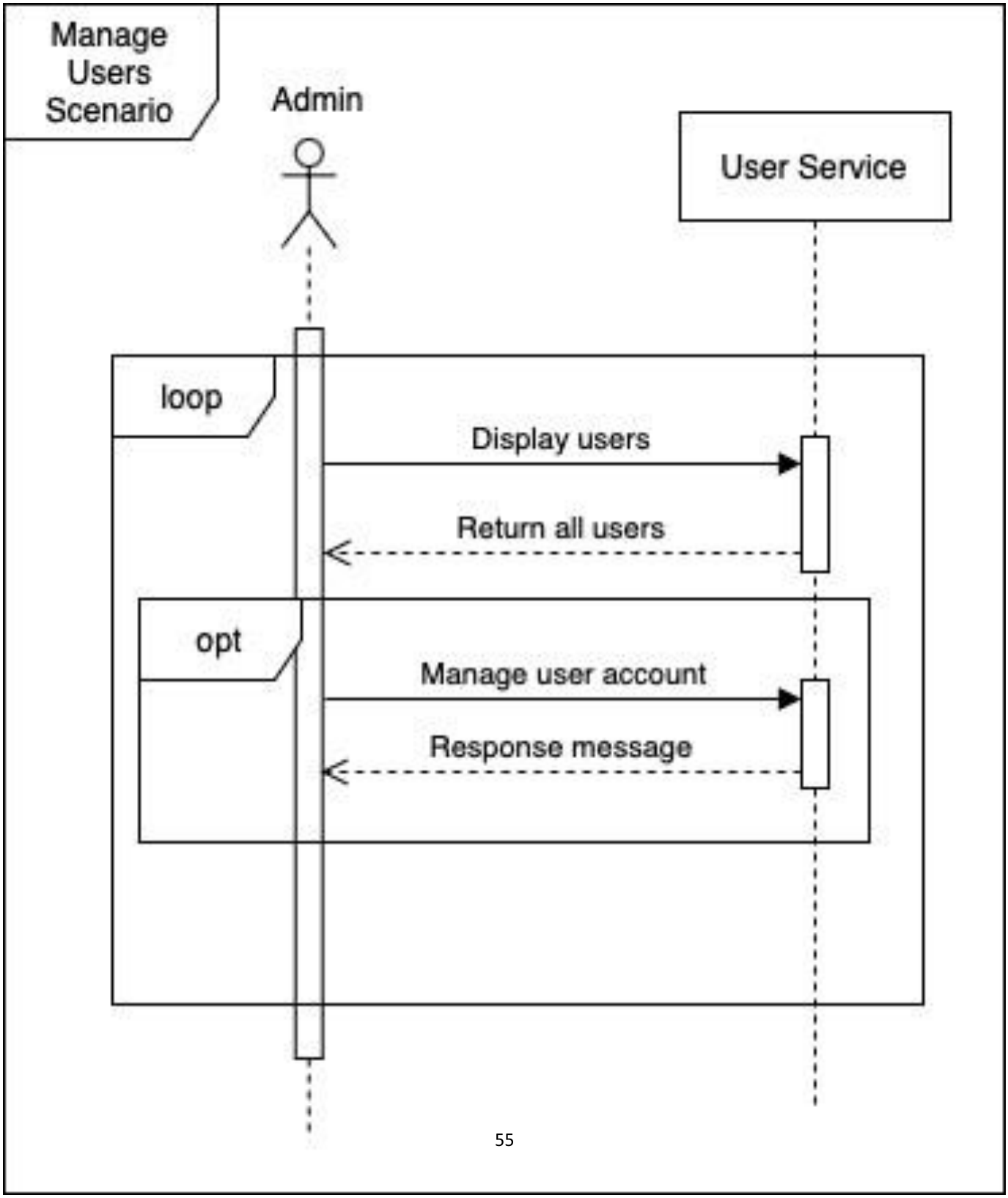

Manage Users Scenario
Admin
User Service
loop
Display users
Return all users
opt
Manage user account
Response message

Figure 22: Manage Users Scenario - Admin Actions on User Accounts

6.9.1 Retrieve List of Users

Administrators can retrieve a paginated list of users using the GET /admin/users endpoint. Optional query parameters allow filtering by name or email. A successful response (HTTP 200) returns user details along with pagination information. Example response:


```
{
  "users": [
    {
      "userId": "user123",
      "name": "John Doe",
      "email": "johndoe@example.com",
      "bio": "I love selling vintage items.",
      "picture": "https://api.remarket.com/images/users/user123.jpg",
      "isSuspended": false,
      "suspendedUntil": null,
      "isBanned": false
    },
    {
      "userId": "user456",
      "name": "Jane Smith",
      "email": "janesmith@example.com", "bio":
      "Handmade crafts seller.",
      "picture": "https://api.remarket.com/images/users/user456.jpg",
      "isSuspended": false,
      "suspendedUntil": null,
      "isBanned": false
    }
  ],
  "page": 1,
  "pageSize": 10,
  "total": 2
}
```


If the admin is not authenticated or does not have the necessary privileges, the service returns 401 Unauthorized or 403 Forbidden errors. Example error response:


```
{
  "message": "Admin privileges required."
}
```


6.9.2 Suspend or Ban a User

To manage a specific user account, such as suspending or banning the user, the administrator interacts with the PUT /admin/users/{userId} endpoint. The request body includes the action (e.g., "suspend" or "ban") and, in the case of suspension, the suspendedUntil date. A successful request (HTTP 200) returns the updated user profile. Example response for a suspension:


```
{
  "userId": "user123",
```

```
  "name": "John Doe",
  "email": "johndoe@example.com",
  "bio": "I love selling vintage items.",
  "picture": "https://api.remarket.com/images/users/user123.jpg",
  "isSuspended": true,
  "suspendedUntil": "2024-12-31T23:59:59Z",
  "isBanned": false
}
```


If the action is invalid, the service returns a 400 Bad Request error. If the specified user is not found, a 404 Not Found error is returned. Example error response:


```
{
  "message": "User not found."
}
```


6.9.3 Error Handling

The User Service ensures only authorized administrators can manage user accounts. Unauthorized access returns 401 Unauthorized, while insufficient privileges return 403 Forbidden. If invalid or missing parameters are provided, a 400 Bad Request error is returned. For example:


```
{
  "message": "Invalid action specified."
}
```


6.9.4 Complete User Management Workflow

The sequence diagram in Figure 22 illustrates the flow of administrative actions, starting with retrieving a list of users, followed by performing specific actions such as suspending or banning accounts. These operations allow administrators to ensure platform compliance and manage user behavior effectively.

6.10 Review Listings Scenario

The review listings scenario allows administrators to manage inappropriate listings on the platform. This includes viewing flagged listings and removing those deemed unsuitable. Figure 23 illustrates the interactions between the Admin and the Listing Service for managing listings.

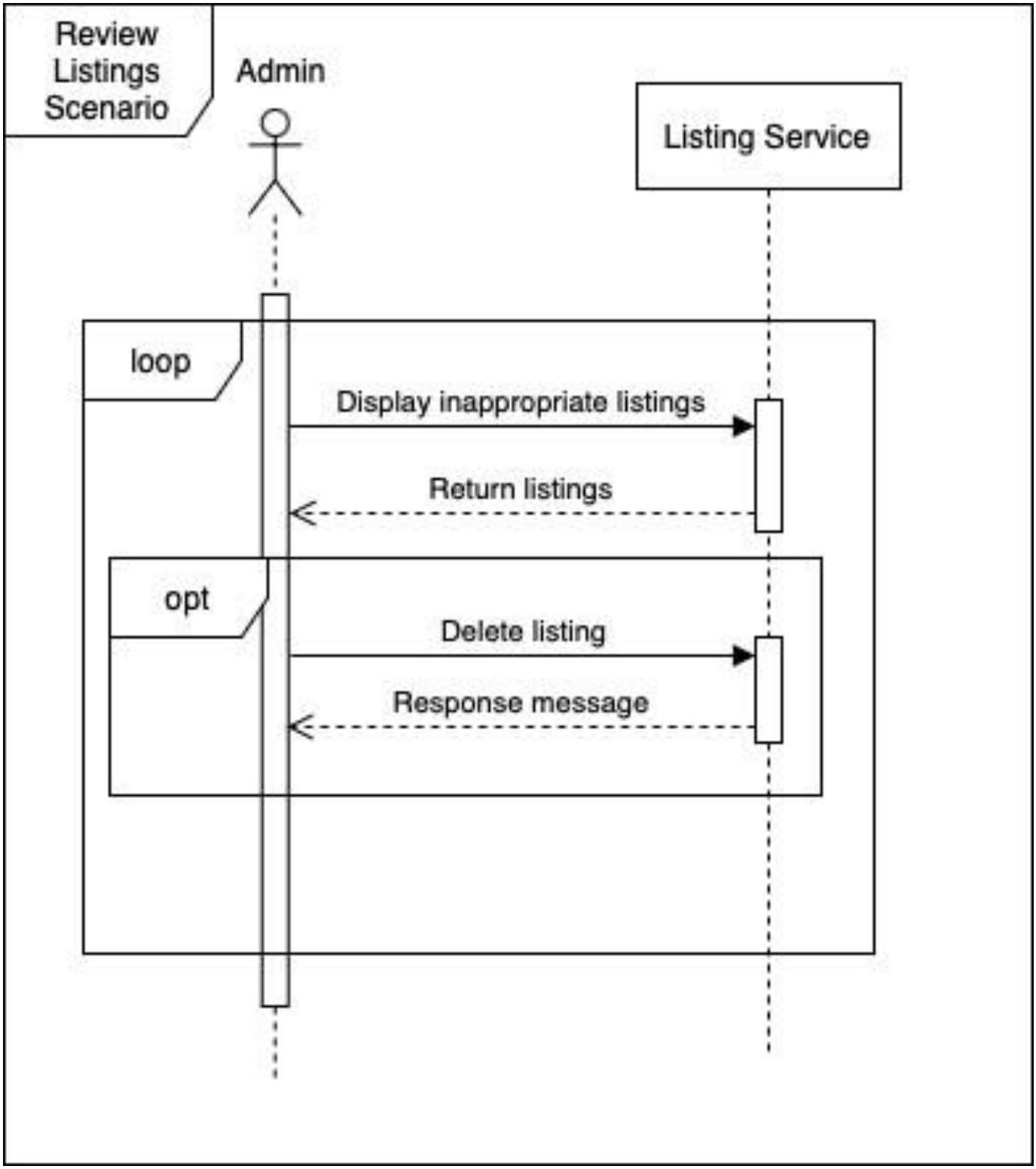

Review Listings Scenario
Admin
Listing Service
loop
Display inappropriate listings
Return listings
opt
Delete listing
Response message

Figure 23: Review Listings Scenario - Admin Actions on Listings

6.10.1 Retrieve Flagged Listings

Administrators can retrieve a list of flagged or inappropriate listings using the GET /admin/listings endpoint. This request supports optional filters for categories or reasons for flagging. A successful response (HTTP 200) returns the flagged listings along with pagination details. Example response:


```
{
  "listings": [
    {
      "id": "listing789",
      "name": "Inappropriate Item",
      "reasonFlagged": "Violates content policy",
      "sellerId": "user123",
      "images": [
        "https://api.remarket.com/images/listings/listing789_1.jpg" ]
    },
    {
      "id": "listing790",
      "name": "Another Inappropriate Item",
      "reasonFlagged": "Spam content",
      "sellerId": "user456",
      "images": [
        "https://api.remarket.com/images/listings/listing790_1.jpg" ]
    }
  ],
  "page": 1,
  "pageSize": 10,
  "total": 2
}
```


If the admin is not authenticated or does not have the necessary privileges, the service returns 401 Unauthorized or 403 Forbidden errors. Example error response:


```
{
  "message": "Admin privileges required."
}
```


6.10.2 Delete a Listing

To remove an inappropriate listing, administrators can use the DELETE /admin/listings/{listingId} endpoint. The listingId parameter identifies the specific listing to delete. A successful deletion returns a 204 No Content status with no response body.

If the listing does not exist or the action is unauthorized, appropriate 404 Not Found or 403 Forbidden errors are returned. Example error response:


```
{
  "message": "Listing not found."
}
```

6.10.3 Error Handling

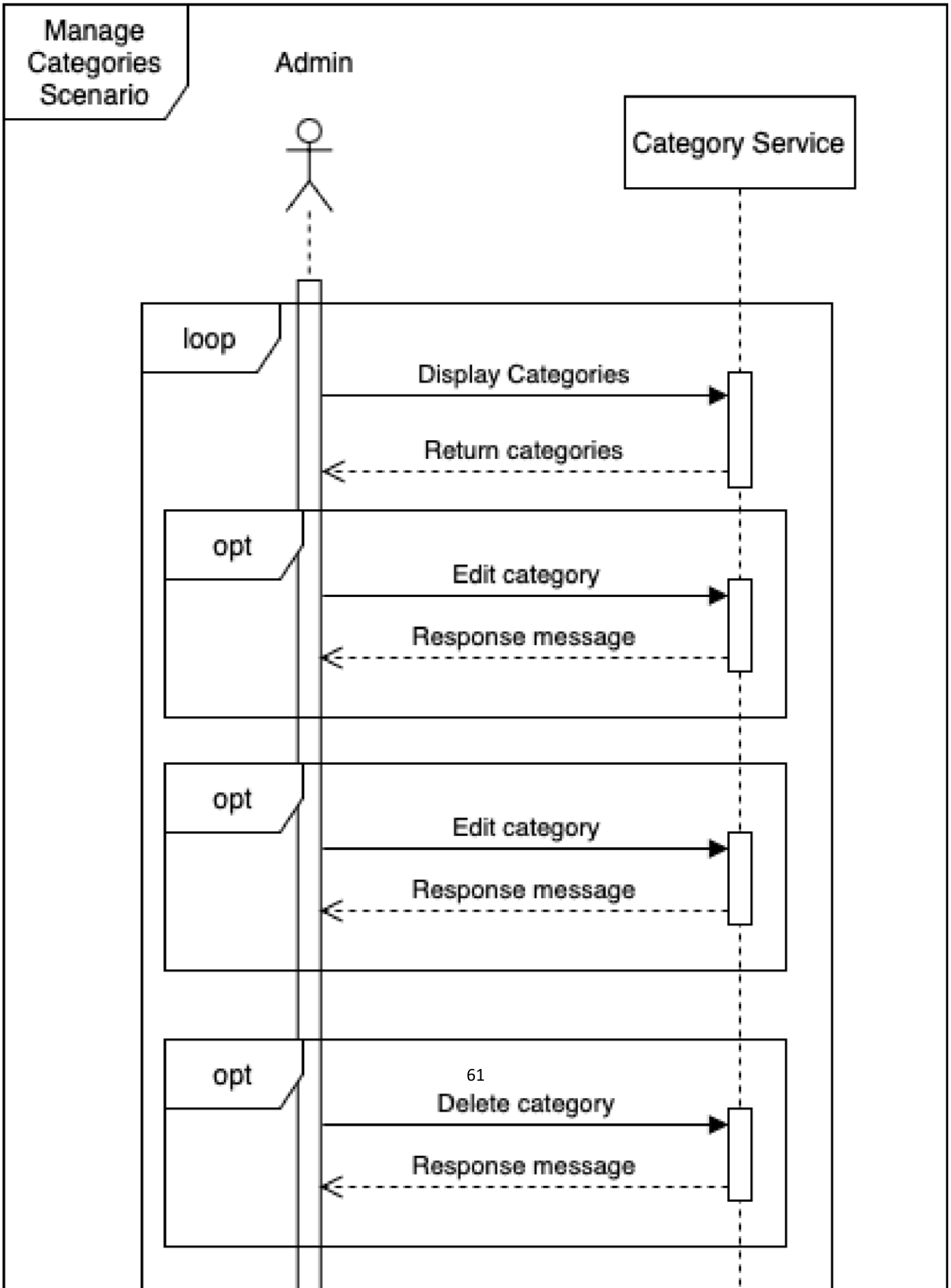
The Listing Service ensures only authorized administrators can delete flagged listings. Any unauthorized access or insufficient privileges result in 401 Unauthorized or 403 Forbidden errors. Invalid or missing parameters result in a 400 Bad Request error.

6.10.4 Complete Review Workflow

The sequence diagram in Figure 23 demonstrates the workflow for reviewing and managing listings. The process begins with retrieving flagged listings, followed by administrative actions such as deleting inappropriate listings. These operations enable administrators to maintain the quality and safety of the platform.

6.11 Manage Categories Scenario

The manage categories scenario allows administrators to create, edit, and delete categories for organizing listings. Figure 24 illustrates the interactions between the Admin and the Category Service for managing categories.


Manage Categories Scenario
Admin
Category Service
loop
Display Categories
Return categories
opt
Edit category
Response message
opt
Edit category
Response message
opt
Delete category
Response message

6.11.1 Retrieve All Categories

Administrators and users can retrieve a list of all available categories using the GET /categories endpoint. A successful response (HTTP 200) returns the list of categories. Example response:


```
{
  "categories": [
    {
      "id": "cat001",
      "name": "Electronics"
    },
    {
      "id": "cat002",
      "name": "Jewelry"
    }
  ]
}
```


6.11.2 Create a New Category

Administrators can create new categories using the POST /categories endpoint. The request payload includes the name of the new category. A successful request (HTTP 201) returns the newly created category. Example response:


```
{
  "id": "cat003",
  "name": "Books"
}
```


If the category name already exists, the service responds with a 400 Bad Request. If the admin is not authenticated or lacks necessary privileges, the service returns 401 Unauthorized or 403 Forbidden errors.

6.11.3 Edit a Category

Administrators can modify existing categories using the PUT /categories/{categoryId} endpoint. The request payload includes the updated category name. A successful request (HTTP 200) returns the updated category. Example response:


```
{
  "id": "cat001",
  "name": "Updated Category Name"
}
```


If the category does not exist, the service responds with a 404 Not Found error. Invalid data results in a 400 Bad Request, while unauthorized or forbidden access returns 401 Unauthorized or 403 Forbidden.

6.11.4 Delete a Category

Administrators can delete a category using the DELETE /categories/{categoryId} endpoint. A successful deletion returns a 204 No Content response.

If the category does not exist, the service responds with a 404 Not Found error. Unauthorized or forbidden access results in 401 Unauthorized or 403 Forbidden errors.

6.11.5 Complete Category Management Workflow

The sequence diagram in Figure 24 demonstrates the workflow for managing categories. The process starts with retrieving categories, followed by creating, editing, or deleting categories as needed. These actions ensure that the platform's categorization system remains organized and up-to-date.

6.12 Review Transactions Scenario

The review transactions scenario enables administrators to monitor and flag transactions that are potentially suspicious or problematic. Figure 25 illustrates the interactions between the Admin and the Order service for reviewing and managing transactions.

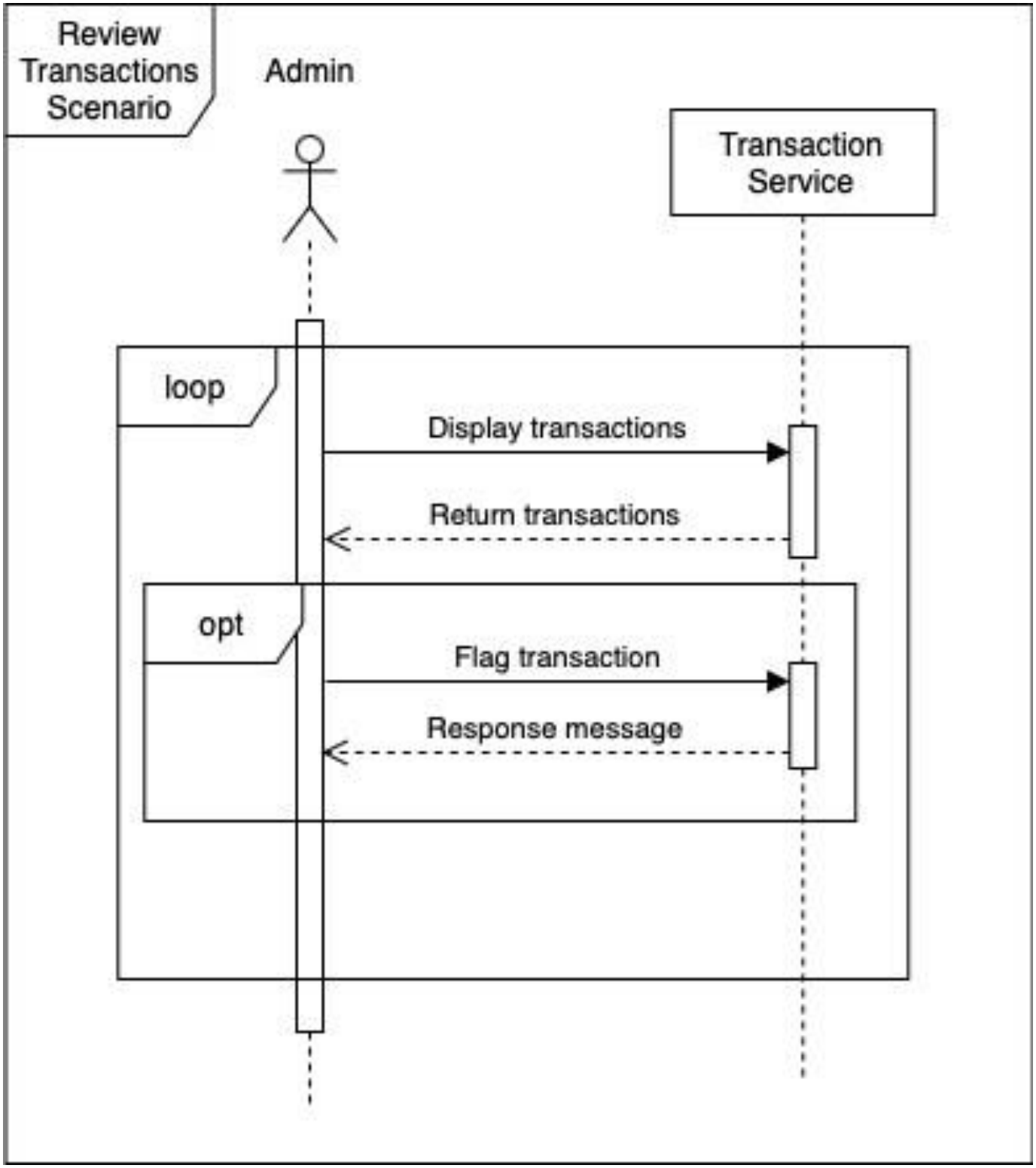

Review Transactions Scenario
Admin
Transaction Service
loop
Display transactions
Return transactions
opt
Flag transaction
Response message

Figure 25: Review Transactions Scenario - Admin Actions on Transactions

6.12.1 Retrieve All Transactions

Administrators can retrieve a paginated list of all transactions using the GET /admin/orders endpoint. Optional query parameters allow filtering by order status or user ID. A successful response (HTTP 200) returns transaction details along with pagination information. Example response:


```
{
  "orders": [
    {
      "orderId": "order001",
      "status": "Processing",
      "totalAmount": 240.00
    }
  ],
  "page": 1,
  "pageSize": 10,
  "total": 1
}
```


If the admin is not authenticated, the service returns 401 Unauthorized or 403 Forbidden errors. Example error response:


```
{
  "message": "Admin privileges required."
}
```


6.12.2 Flag a Transaction

Administrators can flag transactions that require further investigation using the PUT /admin/orders/{orderId} endpoint. The request payload specifies the action to take, such as canceling the order. A successful response (HTTP 200) includes the updated transaction details. Example response:


```
{
  "orderId": "order001",
  "items": [
    {
      "listingId": "listing789",
      "name": "Vintage Camera",
      "price": 150.00,
      "quantity": 1
    },
    {
      "listingId": "listing790",
      "name": "Handmade Necklace",
      "price": 45.00,
      "quantity": 2
    }
  ],
  "totalAmount": 240.00,
```

```
  "status": "Cancelled"
}
```


If the specified order does not exist, the service responds with a 404 Not Found error. Invalid actions or parameters result in a 400 Bad Request error. Example error response:


```
{
  "message": "Invalid action specified."
}
```


6.12.3 Error Handling

The Transaction Service ensures only authorized administrators can manage transactions. Unauthorized or forbidden access results in 401 Unauthorized or 403 Forbidden errors. Missing or invalid data leads to 400 Bad Request responses.

6.12.4 Complete Transaction Review Workflow

The sequence diagram in Figure 25 demonstrates the workflow for reviewing transactions. The process starts with retrieving transactions, followed by optional actions such as flagging or canceling problematic transactions. These administrative actions ensure the integrity and reliability of the transaction system.