CHECKPOINT 2 CS-UH 3260: Software Architecture

Retail Management System Refund and Audit Prepared by:

Terezia Juras (tj2286)
Izah Sohai (is2587)

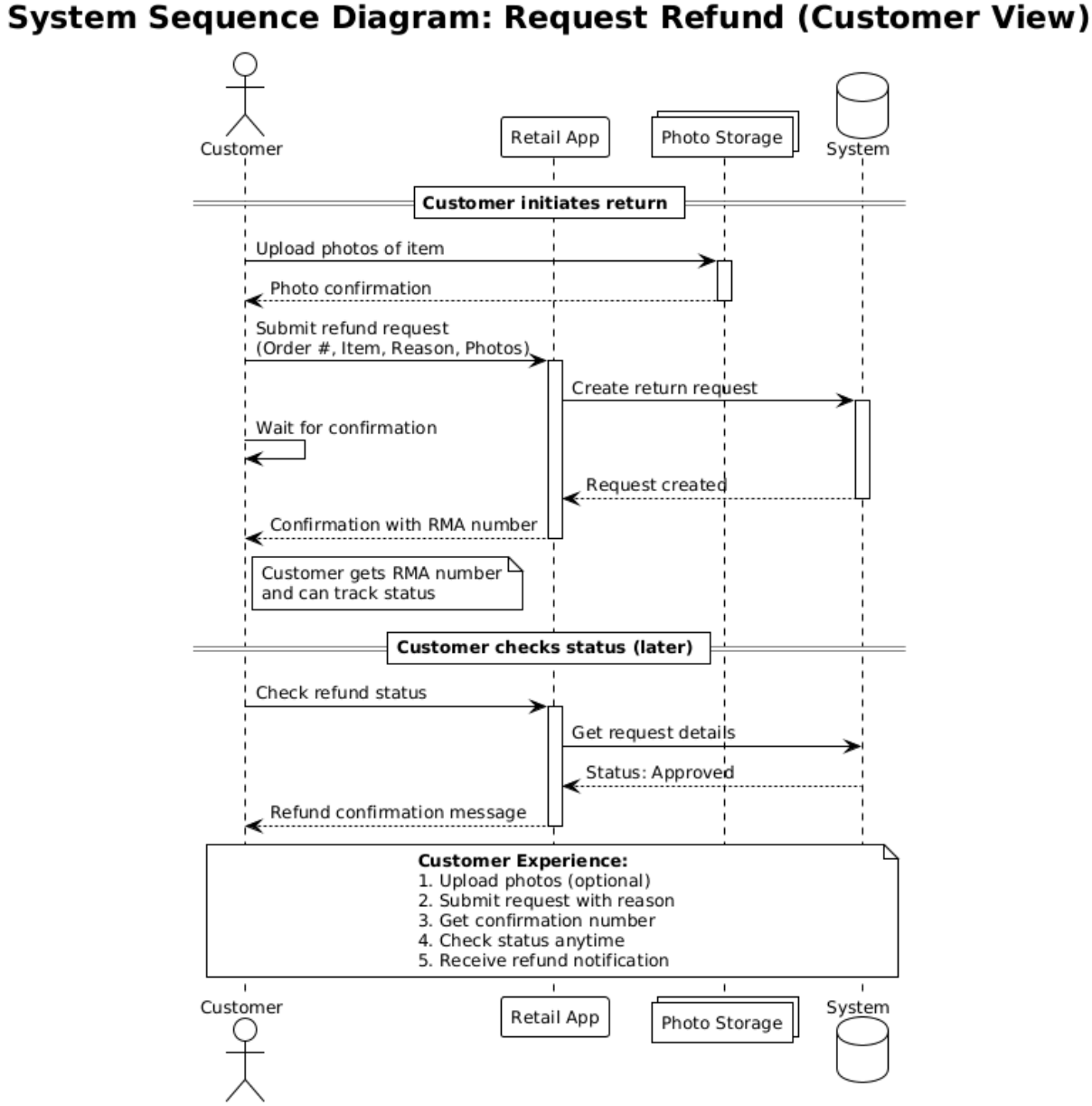

Figure 1: Falcon Market System Sequence Diagram Request Refund (customer view)

First Use Case: Request Refund (Customer View)

Primary Actor

- Customer (authenticated end user)

Supporting Actors

- Photo Storage Service (for item photos)
- Falcon Market System (order and refund backend)
- Retail App (Falcon Market web/mobile front end)

Trigger

- Customer chooses **“Request Refund”** for an item in a past order.

Preconditions

- Customer is logged into the Falcon Market app.
- At least one completed order exists in the customer’s account.
- The selected order item is eligible to be returned according to policy (e.g., within return window).

Postconditions

- A refund/return request exists in the system, linked to the original order and item.
- The request has a unique **RMA number** and an initial status (e.g., “Pending Review”).
- The customer can later use the app to **check the refund status** and see the outcome.

Main Success Scenario (Basic Flow) Request Refund (Customer View)

Part 1 – Customer submits refund request

1. **Customer opens the return flow**
    - From My Orders, the customer selects a completed order and chooses “Request Refund/Return” for a specific item.
2. **Customer uploads photos of the item (optional)**
    - The app prompts the customer to upload photos of the item/packaging.
    - The customer selects 1–N images and confirms upload.

- The app sends the images to the Photo Storage service.
- The Photo Storage service returns a confirmation and references (URLs/IDs) for the uploaded photos.

3. **Customer fills in the refund form**
    - The app shows a refund form with:
        - Selected order and item
        - Reason for return (e.g., “Defective”, “Not as expected”, etc.)
        - Optional description/comments
        - Attached photo references (if uploaded)
    - The customer reviews and confirms the details.
4. **Customer submits the refund request**
    - The customer clicks “Submit Refund Request”.
    - The app validates that required fields (order, item, reason) are present.
5. **App sends refund request to backend system**
    - The app calls the System to create a return/refund request, including:
        - Customer ID
        - Order ID and item details
        - Reason and description
        - Photo references (if any).
6. **System creates return request and assigns RMA**
    - The System validates the request (order belongs to customer, item is in the order, within policy, etc.).
    - On success, the System:

        - Creates a new return/refund record
        - Assigns a unique RMA number
        - Initializes status (e.g., “Pending Review”).
    - The System responds to the app with the new request details, including RMA and status.

7. **App confirms submission to the customer**
    - The app shows a success message with the RMA number (e.g., “Your refund request RMA-12345 has been submitted.”).
    - The app may send an email and/or push notification with the RMA and a summary of the request.
    - The refund request now appears under the customer’s Orders/Returns section with its current status.

Part 2 – Customer checks refund status (extension of same use case)

8. **Customer opens refund status view (later)**
    - At a later time, the customer navigates to My Orders → Returns/Refunds and selects the previously created RMA (e.g., RMA-12345).
    - Alternatively, the customer taps a “Check refund status” action from a notification.

9. **App requests refund status from System**

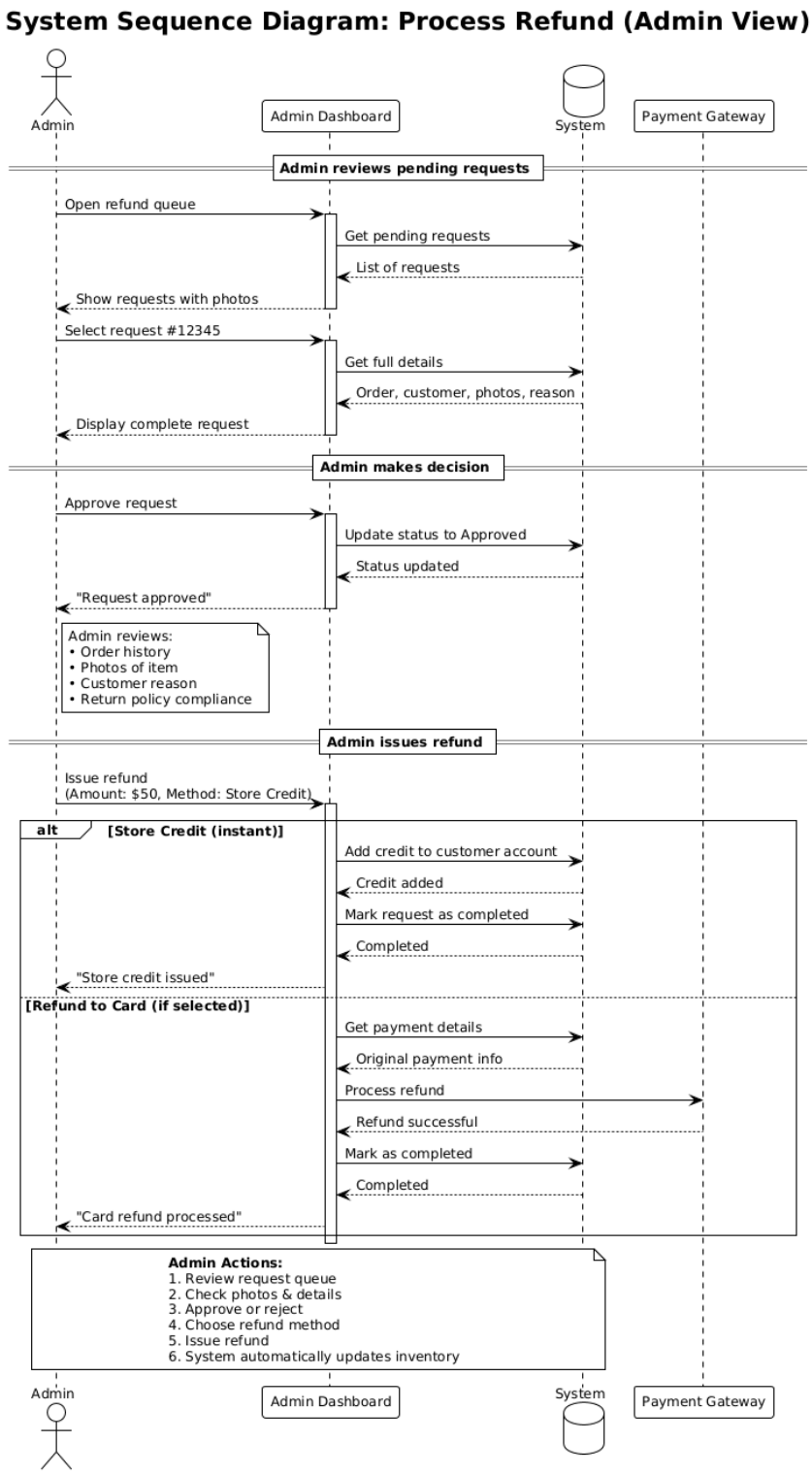

Figure 2: Falcon Market System Sequence Diagram Process Refund (Admin View)

Second Use Case: Process Refund (Admin View)

Primary Actor

- Admin

Supporting Actors

- Admin Dashboard (UI)
- Backend System
- Payment Gateway (for card refunds)

Trigger

- Admin opens the refund queue to handle pending refund requests.

Preconditions

- Admin is authenticated and authorized to manage refunds.
- One or more refund requests exist in a pending state.

Postconditions

- The selected refund request is moved to a final state (e.g., completed after refund is issued).
- A refund is applied through the chosen method (credit or original payment).
- The system state (orders, balances, inventory) is updated consistently.

Main Success Scenario (Basic Flow – Approve and Refund)

1. **Admin opens refund queue**
    - Admin navigates to the Admin Dashboard and opens the list of pending refund requests.

- The Dashboard requests the pending items from the System and displays them.

2. **Admin selects a specific refund request**
    - Admin chooses one request from the list.
    - The Dashboard asks the System for detailed information about that request (order, items, reason, photos, and customer data).
    - The Dashboard presents the full details for review.
3. **Admin reviews the request**
    - Admin inspects the order history, customer explanation, and any photos, and considers applicable return policies.
4. **Admin decides to approve the request**
    - Admin chooses an action to approve the refund request in the Dashboard.
    - The Dashboard instructs the System to update the request status to an approved state.
    - The System confirms the status update and the Dashboard reflects this to the Admin.
5. **Admin chooses refund method and amount**
    - Admin selects a refund method (e.g., store credit or refund to original payment) and confirms the amount to be refunded using the Dashboard.
6. **System issues refund – Store credit path**
    - If **store credit** is selected:
      6.1. The Dashboard asks the System to add credit to the customer's account and link it to the refund.
      6.2. The System applies the credit and confirms back to the Dashboard.
      6.3. The Dashboard instructs the System to mark the request as completed.
      6.4. The System marks the request as completed and confirms.
      6.5. The Dashboard shows the Admin that the store-credit refund flow has finished.
7. **System issues refund – Card refund path**

- If **card refund** is selected instead:
  7.1. The Dashboard requests original payment details from the System.
  7.2. The System returns the relevant payment information.
  7.3. The Dashboard calls the Payment Gateway to initiate the refund.
  7.4. The Payment Gateway confirms that the refund operation succeeded.
  7.5. The Dashboard instructs the System to mark the request as completed.
  7.6. The System marks the request as completed and confirms.
  7.7. The Dashboard shows the Admin that the payment refund flow has finished.

8. **Use case ends**

   - The refund request now has a final status, and associated data (refund record, credit or payment transaction, and any stock adjustments handled by the System) are in a consistent state.

Extensions

E1 – Request deemed not acceptable

- After Step 3, Admin may choose to reject the request instead of approving it.
- The Dashboard instructs the System to set the request to a rejected state.
- The System updates the status and confirms, and the Dashboard reflects the final state to the Admin.

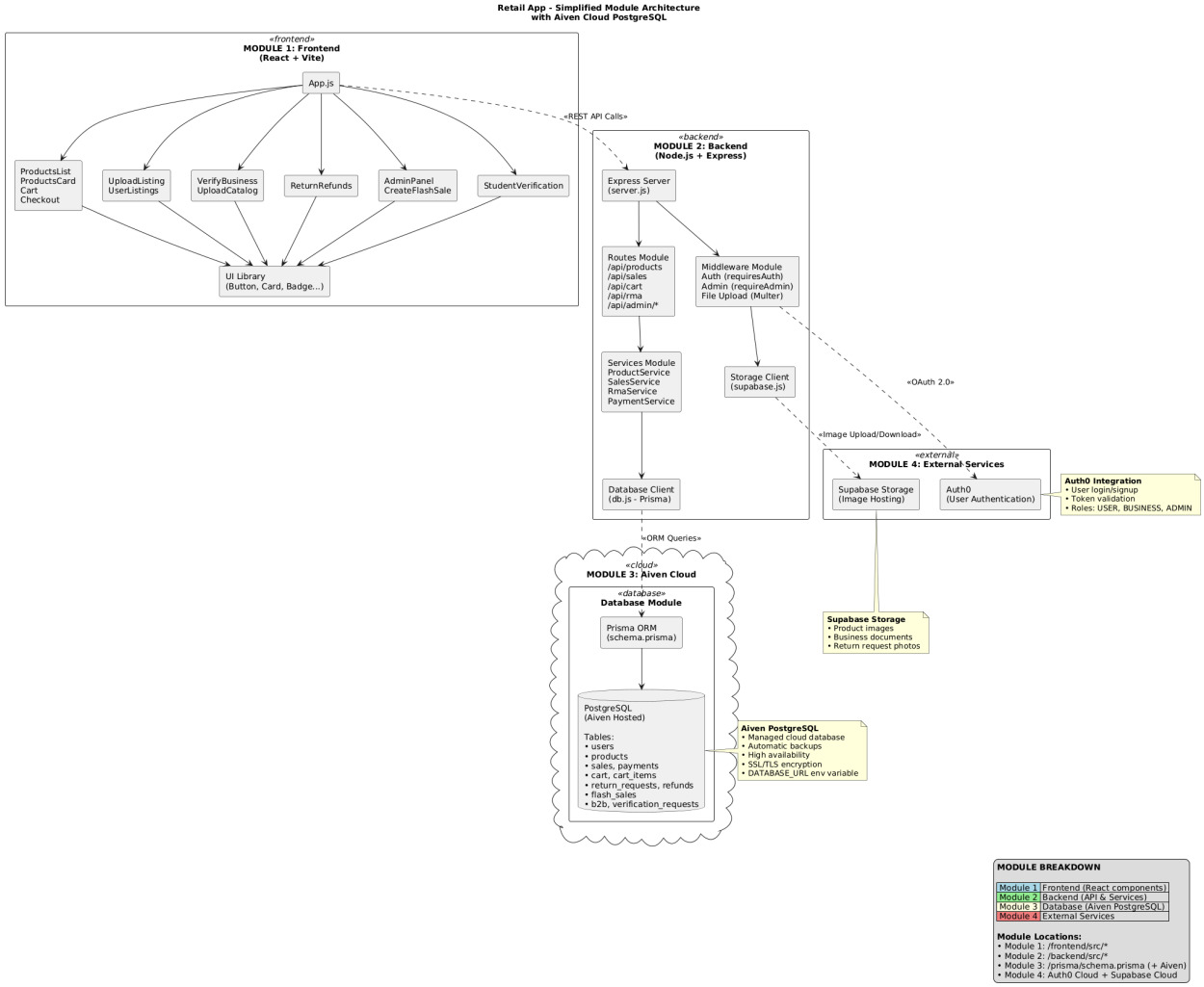

Figure 2: Falcon Market Module Diagram of Frontend, Backend, Database and External Services

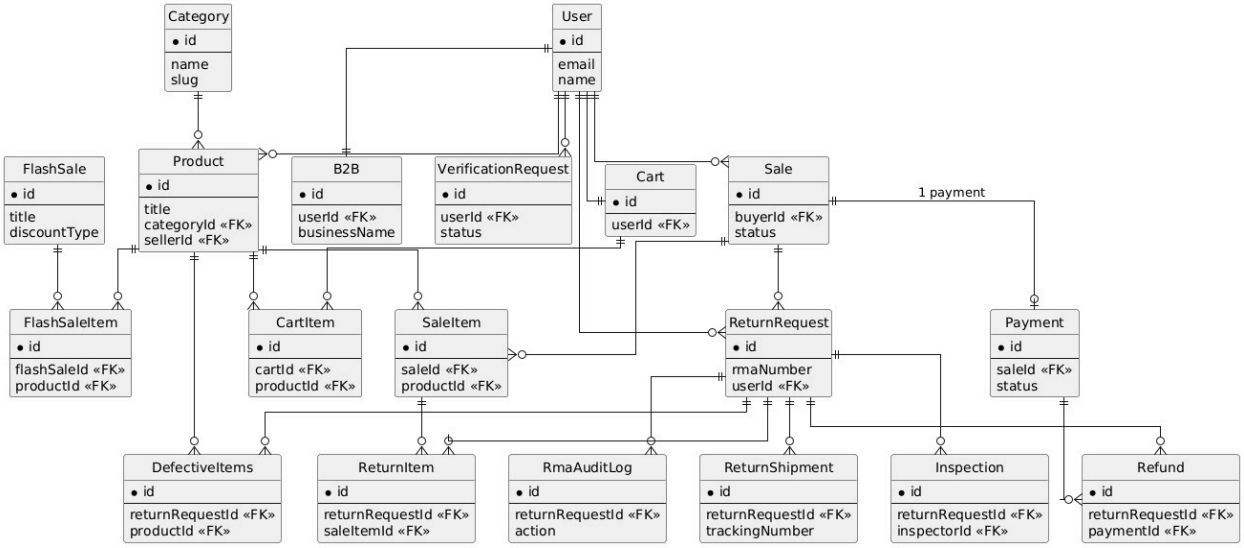

Figure 3: ER Diagram

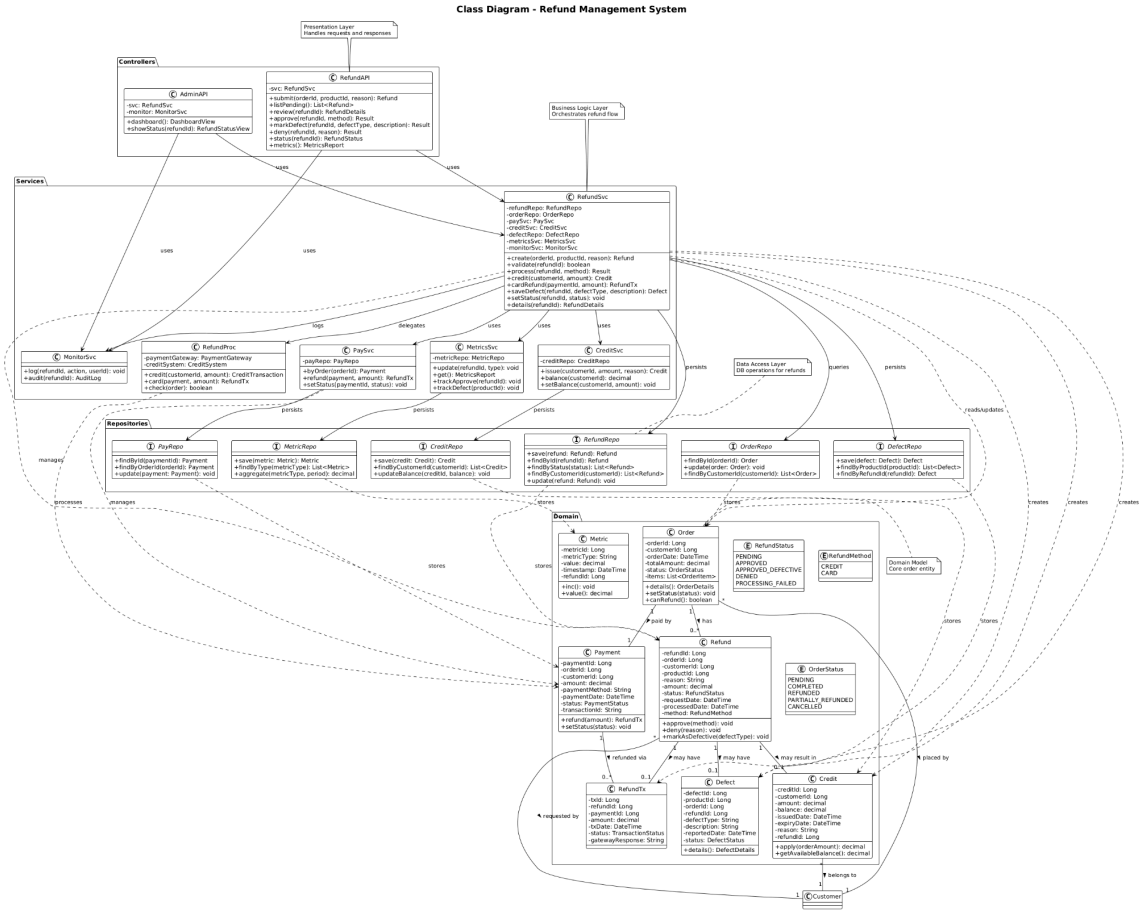

Figure 4: Class Diagram for Refund

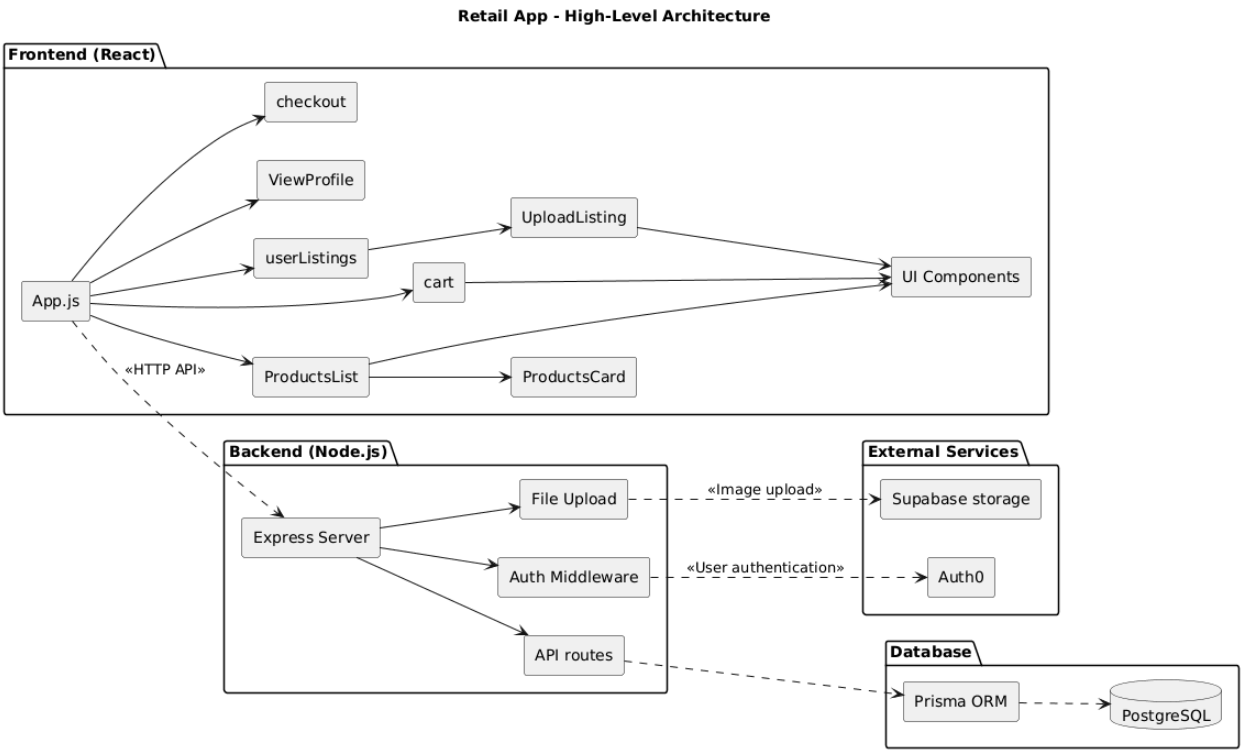

Figure 5: Deployment Diagram