Functional Requirements (FRs) for Payment Gateway Integration

1. Initiation of Payment by User:
    a. The system shall allow the user (buyer) to select one or more products and proceed to the payment page, by providing a "Pay Now" button.
    b. The system shall display the total amount to be paid, including taxes and shipping (if any).

2. Creation of Order in Payment Gateway:
    a. The system shall generate a unique order ID using the Payment Gateway API (i.e. Razorpay) when the user proceeds to pay.
    b. The system shall send the following details to the gateway when creating an order:
        - Amount to be paid
        - Currency
        - Receipt or internal order reference number
    c. The system shall store the generated order ID in the database.

3. Payment Checkout UI:
    a. The system shall render the Checkout form hosted by the payment gateway when the user proceeds to payment.
    b. The Checkout form shall display payment options such as UPI, credit/debit card, net banking, wallets, and pay on delivery, based on seller requirements.
    c. The user shall enter payment details securely on the rendered checkout form.

4. Payment Confirmation Handling:
    a. The system shall receive a payment confirmation (success/failure) from the Payment Gateway via callback/webhook.
    b. On receiving a successful payment confirmation, the system shall verify the payment authenticity using the payment signature provided by the gateway.
    c. The system shall mark the order status as **Paid** in the database after successful verification and direct the user to 'Success' page on marketplace platform.
    d. The system must automatically generate and send a digital receipt to the user upon a successful payment, which should include the order details and a unique transaction ID.
    e. On payment failure or cancellation, the system shall mark the order status as **Failed** or **Cancelled** and display an appropriate message to the user.

5. Order Status Update:
    a. The system shall store payment details in the database, including:
        - Payment ID
        - Order ID
        - Payment Status (Success, Failed, Cancelled)
        - Timestamp of the transaction
        - Payment Mode (UPI, Card, Net Banking, etc.)
    b. The system shall allow admin and users to view the status of payments for orders.

6. Error and Retry Handling:
    a. The system shall provide a mechanism for the user to retry payment if the payment failed due to a temporary error.

b. The system shall notify the user about reasons for failure when possible (e.g., insufficient funds, network error).

7. Logging and Auditing:
    a. The system shall log every payment attempt (successful or failed) along with timestamps for audit purposes.
    b. The system shall log payment signature verification results to help debug payment disputes.

8. Post-Payment Functionality:
    a. The system must provide a mechanism for administrators or sellers to initiate a full or partial refund for a successful payment via the integrated Payment Gateway.
    b. The system must manage the settlement of funds from the Gateway into the business's bank account (marketplace owner/administrator).

Non-Functional Requirements (NFRs)

1. Payment initiation and confirmation shall be processed within 5-10 seconds under normal network conditions.

2. The payment system shall ensure at least 99.9% uptime.

3. All communication with the payment gateway shall use HTTPS with TLS 1.2+ encryption.

4. The system shall never store sensitive cardholder data such as card numbers, CVV, or PIN.

5. API keys and secrets shall be stored securely in environment variables or encrypted config files.

6. The payment checkout interface shall be mobile-friendly and responsive.

7. Clear success, failure, or cancellation messages shall be displayed in plain language.

8. All payment attempts shall be logged with order ID, payment ID, status, and timestamp, excluding sensitive data.

9. The system shall allow retry of failed payments without losing transaction context.

10. The integration shall comply with PCI-DSS and applicable financial regulations (via the respective payment gateway compliance, like Razorpay in our case).

11. Payment gateway SDK and API versions shall be kept updated to the latest stable release.

12. The system shall scale to handle peak loads up to 2x the average transaction volume.