

as of now, even though it is multi-seller platform, the payment process is same as single merchant. how exactly can we split the payment for different sellers, and additonaly(optionally) how can we allocate a commision for super admin/platform fee? we use razorpay . give a high level overview first. guide me, i want to do it myself

♦ Of course. I can certainly guide you through this. Implementing a marketplace payment solution is a significant architectural change, so it's wise to start with a high-level overview.

The feature you're looking for from Razorpay is called Razorpay Route. It is specifically designed for marketplaces and platforms like yours to split payments between multiple sellers (vendors) and take a commission.

Here is a high-level overview of how it works and the changes you'll need to make.

#### High-Level Overview: From Single-Merchant to Marketplace

##### Your Current Flow (Single-Merchant Model):

1. Customer Pays: A customer pays the total cart amount.
2. Payment to You: The entire amount is captured into your single, primary Razorpay account.
3. Manual Payouts: You are responsible for manually calculating how much each seller earned and transferring their money to them later (e.g., via bank transfer). This is complex, slow, and hard to scale.

##### The New Flow (Razorpay Route Marketplace Model):

1. Seller Onboarding:
  - \* You create "Linked Accounts" (or "Vendor Accounts") for each of your sellers through the Razorpay API.
    - \* This process involves collecting the seller's bank account details and identity information (KYC). Razorpay manages the verification.
    - \* You store the unique account\_id for each seller in your database, associated with their seller profile.
2. Customer Checkout:
  - \* The customer's checkout experience remains largely the same. They see a total cart value and pay for it.
  - \* The entire payment is initially credited to your main platform Razorpay account.
3. Automatic Payment Splitting (The "Route" Logic):
  - \* This is the core change. After the payment is successfully captured by your account, your backend server immediately initiates transfers using the Razorpay Route API.
  - \* Your server calculates:

- \* Which products in the order belong to which seller.
- \* The amount to be transferred to each seller.
- \* The commission/platform fee you want to retain.
- \* You then make API calls to "transfer" the specific amounts to the respective sellers' Linked Accounts.

### Example Scenario

Imagine a customer buys two items:

- \* Product A: ₹1,000 (from Seller A)
- \* Product B: ₹500 (from Seller B)
- \* Your Platform Commission: 10%

The process would be:

1. Customer pays the total of ₹1,500.
2. The full ₹1,500 lands in your primary Razorpay account.
3. Your backend code is triggered by the successful payment. It calculates:
  - \* Seller A's share: ₹1,000 - 10% (₹100) = ₹900
  - \* Seller B's share: ₹500 - 10% (₹50) = ₹450
  - \* Your total commission: ₹100 + ₹50 = ₹150
4. Your server makes two API calls to Razorpay:
  - \* POST /transfers - Transfer ₹900 to Seller A's account\_id.
  - \* POST /transfers - Transfer ₹450 to Seller B's account\_id.
5. The remaining ₹150 stays in your Razorpay account automatically. The funds are settled to the respective bank accounts based on Razorpay's settlement cycle.

This automates the entire process, ensuring sellers are paid promptly and you receive your commission on every transaction without manual intervention.

---

This is the conceptual framework. When you're ready, we can proceed to the first step: onboarding sellers and creating Linked Accounts. Shall we move on to that?

> yes. only teach me, and how exactly we will use routes? do we need to enable anything on razorpay side

♦ Excellent questions. You're right to ask about the setup process first.

1. Enabling Razorpay Route on Your Dashboard

Yes, you absolutely need to enable Razorpay Route on your account before you can write any code. It is a special feature that is not active by default.

Action you must take:

1. Log in to your Razorpay Dashboard.
2. Navigate to Settings in the bottom-left menu.
3. Look for a section named Route (Marketplace) or something similar.
4. You will likely see a button to "Request Access" or "Enable".
5. Razorpay will ask for details about your business model. You must explain that you are running a multi-seller e-commerce platform and need to route payments to your vendors.
6. Their team will review your request. This can take a few business days, and they may contact you for more information.

You cannot proceed with the coding steps until Razorpay has approved and enabled Route for your account.

---

## 2. How Exactly Will We Use Routes in the Code?

Once Route is enabled, you'll use it in two main phases within your application. Here's a breakdown of the coding tasks.

### Phase 1: Seller Onboarding (Creating Linked Accounts)

This is a one-time setup for each seller. You'll need a new section in your admin/seller area for this.

- \* What you'll build: A form where a seller can submit their bank account details (Account Number, IFSC, Account Holder Name).
- \* Backend Logic (likely in `helpers/admin-helpers.js` or a new `seller-helpers.js`):
  1. When the form is submitted, your Node.js server receives the seller's details.
  2. You will use the Razorpay Node.js SDK to make an API call to POST /accounts.
  3. This call creates a "Linked Account" and Razorpay returns an account\_id (e.g., acc\_G2iBCB6E8Q3g6A).
  4. Crucially, you must save this `account\_id` in your database, associating it with that seller's document in your users collection (or a dedicated sellers collection).

### Phase 2: Payment Splitting (After a Customer Pays)

This is the main logic that runs after every successful order.

- \* Where to put the code: The best place is inside your payment verification logic. In your routes/users.js, you probably have a route that handles the callback from Razorpay to verify a payment's signature. The split-payment logic goes right after a successful verification.

\* Backend Logic (e.g., in your `/verify-payment` route):

1. Verify the Payment: First, confirm the payment is genuine using Razorpay's signature verification, just as you do now.
2. Get Order and Product Details: Fetch the order details from your database. You need to know which products were in the cart and who the seller is for each product.
3. Calculate Amounts:
  - \* Create an object or map to figure out the total amount owed to each seller from this one order.
  - \* For each seller's total, calculate your commission (e.g., sellerTotal \* 0.10).
  - \* Determine the final transfer amount for each seller (sellerTotal - yourCommission).
4. Create the Transfer: This is the core Razorpay Route step. You will make a single API call to POST /payments/{payment\_id}/transfers.

This call contains an array of all the transfers to be made from that one payment.

Example using the Razorpay SDK:

```
1 // This code runs after you've verified the payment
2 // and calculated the amounts for Seller A and Seller B.
3
4 // Let's assume paymentId is from the webhook/callback
5 const paymentId = 'payXXXXXXXXXXXXXX';
6
7 // Let's assume you've fetched these from your database
8 const sellerA_accountId = 'accXXXXXXXXXXXXXX';
9 const sellerB_accountId = 'accYYYYYYYYYYYYYYYY';
10
11 // Amounts must be in the smallest currency unit (e.g., paise for INR)
12 const sellerA_transferAmount = 900 * 100; // 900 INR
13 const sellerB_transferAmount = 450 * 100; // 450 INR
14
15 try {
16   const transfer = await razorpay.payments.transfer(paymentId, {
17     transfers: [
18       {
19         account: sellerA_accountId,
```

```
20     amount: sellerA_transferAmount,
21     currency: "INR",
22   },
23   {
24     account: sellerB_accountId,
25     amount: sellerB_transferAmount,
26     currency: "INR",
27   },
28 ],
29 });
30 console.log("Transfers successful:", transfer);
31 // Now, update your order status to "Completed" or "Processing"
32 } catch (error) {
33   console.error("Transfer failed:", error);
34   // IMPORTANT: Handle this failure. You might need to retry later.
35 }
```

The amount that you don't transfer away automatically remains in your account as your commission.

Your first step is to get Route enabled on your Razorpay dashboard. Let me know once you have confirmation from them, and we can start architecting the code changes.