# **Project Roadmap: Stripe Integration for Hotel Subscriptions & Orders**

## ****1. Stripe Account Setup****

**Task:** Configure Stripe for Super Admin and hotel Connect accounts.  
**Details:**

* Super Admin account → receives subscription payments from hotels.
* Enable Stripe Connect → for hotels to receive payments from customers.
* Obtain Stripe keys: STRIPE\_KEY, STRIPE\_SECRET, STRIPE\_CLIENT\_ID (Connect).
* Set up webhook URL in Stripe Dashboard: /api/stripe/webhook.

## ****2. Hotel Stripe Onboarding (Connect)****

**Task:** Allow hotels to connect their Stripe accounts.  
**API Endpoint:**

GET /api/hotel/{hotel}/connect-stripe

GET /api/stripe/callback

**Request Example:**

* Hotel clicks "Connect Stripe" → redirected to Stripe OAuth.
* Stripe redirects back with code → use to fetch stripe\_account\_id.

**Response Example:**

{

"message": "Stripe connected successfully!",

"stripe\_account\_id": "acct\_1Example"

}

## ****3. Subscription Plans Management****

**Task:** Manage internal subscription plans in the system.  
**Notes:**

* Plans are stored in plans table: name, amount, interval (month, quarter, year).
* Frontend fetches available plans for hotels.
* Stripe **does not store these plans**; payment amount is taken directly from DB.

## ****4. Hotel Subscription API****

**Task:** Hotels subscribe to a plan → payment goes to Super Admin.  
**API Endpoint:**

POST /api/hotel/{hotel}/subscribe

**Request Example:**

{

"plan\_id": 2,

"payment\_method\_id": "pm\_ABC123" // Stripe payment method

}

**Response Example:**

{

"payment\_intent\_id": "pi\_ABC123",

"status": "requires\_confirmation",

"plan": "Monthly"

}

**Backend Flow:**

1. Fetch plan amount and interval from DB (plans table).
2. Create Stripe **PaymentIntent** for that amount → money goes to Super Admin.
3. Confirm payment on frontend using Stripe.js.
4. After success, update hotels.subscription\_status = active and hotels.current\_plan.

## ****5. Subscription Upgrade / Downgrade****

**Task:** Hotels can change plans.  
**API Endpoint:**

POST /api/hotel/{hotel}/subscription/change

**Request Example:**

{

"new\_plan\_id": 3,

"payment\_method\_id": "pm\_DEF456"

}

**Response Example:**

{

"payment\_intent\_id": "pi\_DEF456",

"status": "requires\_confirmation",

"plan": "Quarterly"

}

**Backend Flow:**

* Calculate difference between old plan and new plan (optional for prorated charge).
* Create new Stripe PaymentIntent for upgrade/downgrade amount.
* Update current\_plan and subscription\_status in DB after payment success.

## ****6. Customer Order Payment API****

**Task:** Customers pay for hotel services → money goes to hotel account.  
**API Endpoint:**

POST /api/orders

**Request Example:**

{

"hotel\_id": 5,

"amount": 200,

"payment\_method\_id": "pm\_ABC123"

}

**Response Example:**

{

"order\_id": 101,

"payment\_status": "paid"

}

**Backend Flow:**

* Use Stripe **PaymentIntent** with transfer\_data.destination = hotel’s stripe\_account\_id.
* Update orders table with stripe\_payment\_id and payment\_status.

## ****7. Webhook Handling****

**Task:** Listen to Stripe events → update subscription and order status.  
**API Endpoint:**

POST /api/stripe/webhook

**Events to Handle:**

* payment\_intent.succeeded → mark subscription or order as paid.
* payment\_intent.payment\_failed → mark subscription or order as failed.

**Security:**

* Verify webhook signature using STRIPE\_WEBHOOK\_SECRET.

## ****8. Frontend Integration****

**Task:**

* Display available subscription plans (from DB).
* Allow hotel to connect Stripe account.
* Process subscription payments → show success/failure messages.
* Process customer orders → redirect to Stripe payment or confirm status.

## ****9. Testing & Verification****

**Task:**

* Test all internal plans for new subscriptions and upgrades/downgrades.
* Test customer payments to hotels via Stripe Connect.
* Test webhook events and verify DB updates.
* Test edge cases: failed payments, canceled subscriptions, partial upgrades.

## ****Optional Enhancements****

* Send email notifications for successful subscriptions or order payments.
* Add admin dashboard for monitoring hotel subscriptions, revenue, and orders.
* Implement trial periods or discounts for internal plans.