Recurring pricing models

Learn about common pricing models and how to create them.

Pricing models are patterns that represent your business on Stripe. With Product API and Price API objects, you can model what you sell and how you charge for it.

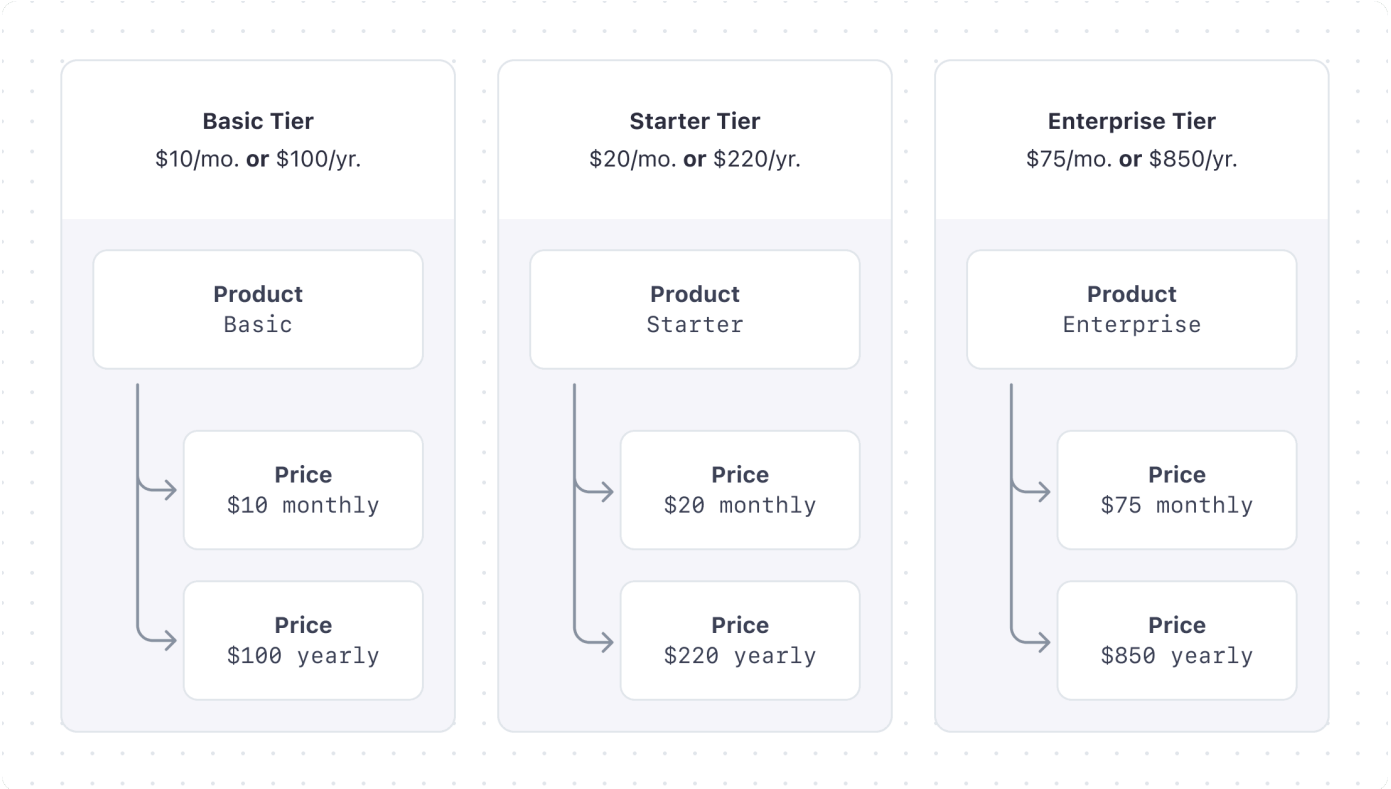
Flat rate: Good-better-best

Many SaaS businesses offer their customers a choice of escalating service options. This flat-rate pricing model is called *good-better-best*. Customers choose a service tier (good, better, or best) and pay a flat rate for it.

Imagine a business called Togethere that sells a collaboration platform. They offer three different service levels: basic, starter, and enterprise. For each service level, they offer a monthly and yearly price. Togethere operates in several countries, so they have prices in multiple currencies.

In this example, Togethere has three products: `Basic`, `Starter`, `Enterprise`. Each product has several different prices. The basic level has prices for 10 USD per month and 100 USD per year. Both prices are for the same `Basic` product, so they share the same product description on the customer's receipt and invoice.

Here's what that model looks like on Stripe:



Flat rate: Good-better-best pricing model

Model good-better-best on Stripe

Dashboard | API

To create a good-better-best model on Stripe through the Dashboard follow the steps below.

First, create the `Basic` product. To learn about all the options for creating a product, see the prices guide.

1. Go to Product catalog.
2. Click **+ Add product**.
3. Enter the `Name` of your product.
4. *(Optional)* Add a **Description**. The description appears at checkout, on the customer portal, and in quotes.

Next, create the monthly price for the `Basic` product:

1. Click **Advanced pricing options**.
2. Select **Recurring** and choose **Flat rate** for the pricing model.
3. Enter the price amount-in this case, `10.00`.

4. Select **Monthly** for the **Billing period**.
5. Click **Next** to save the price.

Then, create the yearly price for the `Basic` product:

1. Click **+ Add another price**.
2. Select **Recurring** and choose **Flat rate** for the pricing model.
3. Enter the price amount-in this case, `100.00`.
4. Select **Yearly** for the **Billing period**.
5. Click **Add product** to save the product and price. You can edit the product and price later.

The subscription integration guide explains how to fit pricing models into a full integration.

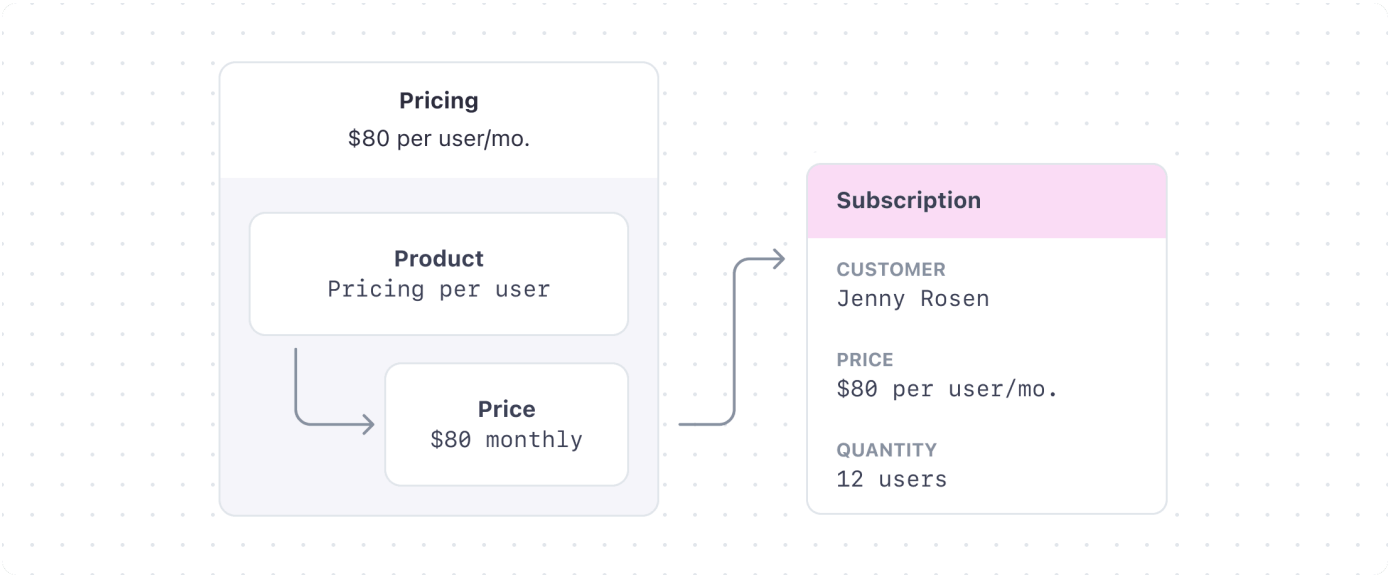
- If you're using Stripe Checkout, the next step is to create a Checkout session for your site. Make sure to set up Stripe.
- If you're using Stripe Elements, the next step is to create a Customer. Make sure you set up Stripe and the sample application.

Per-seat

Togethere, our example collaboration platform company, also wants to offer a per-seat plan. Per-seat pricing is another common offering for SaaS businesses. Togethere's customers pick how many seats they'll use, and Togethere charges based on that amount.

To model this scenario, Togethere creates a product and price structure where each unit represents a user. When Togethere creates a subscription for a customer, the customer specifies the number of users for that subscription.

Here's what that model looks like on Stripe:



See the advanced pricing model section for examples of other models. Flat rate and per-seat pricing, where you charge a customer a flat monthly fee in addition to a per-seat rate, is another common model for SaaS businesses.

Model per-seat pricing on Stripe

Dashboard | API

To create a per-seat model on Stripe through the Dashboard follow the steps below.

First, create the `Per-seat` product. To learn about all the options for creating a product, see the prices guide.

1. Go to Product catalog.
2. Click **+Add product**.
3. Enter the **Name** of the product: `Per-seat`, in this case.
4. *(Optional)* Add a **Description**. The description appears at checkout, on the customer portal, and in quotes.

Next, create the monthly price for the `Per-seat` product:

1. Select **Recurring**.
2. Enter the price amount-in this case, `80.00`.
3. Select **Monthly** for the **Billing period**.
4. Click **Add product** to save the product and price. You can edit the product and price later.

To create a subscription using that price:

1. Go to the **Payments** > **Subscriptions** page.
2. Click **+ Create subscription**.
3. Find or add a customer.
4. Search for the `Per-seat` product you created and select the price you want to use.
5. *(Optional)* Select **Collect tax automatically** to use Stripe Tax.
6. Click **Start subscription** to start it immmediately or **Schedule subscription** to start it on another schedule.

The subscription integration guide explains how to fit pricing models into a full integration.

- If you're using Stripe Checkout, the next step is to create a Checkout session for your site. Make sure you set up Stripe.
- If you're using Stripe Elements, the next step is to create a Customer. Make sure you set up Stripe and the sample application.

For other versions of per-seat pricing, see the advanced models section.

Usage-based pricing

Usage-based pricing is a common pricing model for SaaS businesses that enables you to charge based on a customer's usage of your product or service. As a business, you provide access to your service and bill your customer based on their usage. To see what an end-to-end integration based on the usage-based pricing model looks like, read the usage-based pricing models guide.

Tiered pricing

Prices can represent tiers, allowing the unit cost to change with quantity or usage. Togethere might, for example, want to offer lower rates for customers who use more projects per month. The following examples show two different ways to adjust pricing as usage increases: volume-based pricing and graduated pricing. To demonstrate these approaches to tiered pricing, we'll use the following tiers:

NUMBER OF PROJECTS	PRICE PER TIER
1-5	7 USD
6-10	6.50 USD
11+	6 USD

Use `tiers` if you need non-linear pricing when `quantity` or usage API changes. You can also combine tiered pricing with base fees to create more complex pricing models.

When you create a price API with `billing_scheme=per_unit`, the `unit_amount` is the same regardless of how many units a customer buys. Stripe multiplies this amount by the

`quantity` to determine the total cost. For example, a `unit_amount` of 5 USD creates this billing structure:

QUANTITY/USAGE AT END OF PERIOD	TOTAL COST
1	5 USD
5	25 USD
6	30 USD
20	100 USD
25	125 USD

When you create a price with `billing_scheme=tiered`, the unit cost varies depending on how many units your customer buys. Here's an example tier structure:

TIER	AMOUNT (UNIT COST)
1-5 (`up_to=5`)	5 USD (`unit_amount=500`)
6-10 (`up_to=10`)	4 USD (`unit_amount=400`)
10-15 (`up_to=15`)	3 USD (`unit_amount=300`)
15-20 (`up_to=20`)	2 USD (`unit_amount=200`)
20+ (`up_to=inf`)	1 USD (`unit_amount=100`)

Multiplication happens:

- At the start of the billing period if the `Price` objects have `recurring.usage_type = licensed`.
- At the end of the billing period if the `Price` objects have `recurring.usage_type = metered`.

With tiered billing, you:

- Set the `tiers_mode` to either `volume` or `graduated`.
- Create a `tiers` array to represent the tiers structure.

See also the volume-based and graduated pricing examples.

Volume-based pricing

With volume-based pricing, the subscription item is billed at the tier corresponding to the amount of usage at the end of the period. To implement volume-based billing tiers, set `volume` as the value of `tiers_mode`:

Command Line cURL

```
$  curl https://api.stripe.com/v1/prices \
>    -u "sk_test_4eC39HqLyjWDarjtT1zdp7dc:" \
>    -d nickname="Project Volume Pricing" \
>    -d "tiers[0][unit_amount]"=700 \
>    -d "tiers[0][up_to]"=5 \
>    -d "tiers[1][unit_amount]"=650 \
>    -d "tiers[1][up_to]"=10 \
>    -d "tiers[2][unit_amount]"=600 \
>    -d "tiers[2][up_to]"=inf \
>    -d currency=usd \
>    -d "recurring[interval]"=month \
>    -d "recurring[usage_type]"=metered \
>    -d product={{PRODUCT_ID}} \
>    -d tiers_mode=volume \
>    -d billing_scheme=tiered \
>    -d "expand[0]"=tiers
```

Under volume-based pricing, a customer with 5 projects is charged 35 USD (5 × 7 USD). If they accumulate 6 projects the following month, then all projects are billed at the `6–10` rate. That month, they're charged 39 USD (6 × 6.50 USD).

With `tiers_mode=volume`, the entire `quantity` (or `usage`) is multiplied by the unit cost of the tier.

QUANTITY/USAGE AT END OF PERIOD	UNIT COST	TOTAL FOR VOLUME TIERED PRICING
1	7 USD	7 USD
5	7 USD	35 USD
6	6.5 USD	39 USD
20	6 USD	120 USD
25	6 USD	150 USD

Because the tier price applies to the entire `quantity` (or `usage`), the total may decrease when calculating the final cost.

Graduated pricing

While similar to volume pricing, graduated pricing charges for the usage in each tier instead of applying a single price to all usage. To use graduated tiers, set the value of `tiers_mode` to `graduated`:

```
$  curl https://api.stripe.com/v1/prices \
>    -u "sk_test_4eC39HqLyjWDarjtT1zdp7dc:" \
>    -d nickname="Per-minute pricing" \
>    -d "tiers[0][unit_amount]"=500 \
>    -d "tiers[0][up_to]"=5 \
>    -d "tiers[1][unit_amount]"=400 \
>    -d "tiers[1][up_to]"=10 \
>    -d "tiers[2][unit_amount]"=100 \
>    -d "tiers[2][up_to]"=inf \
>    -d currency=usd \
>    -d "recurring[interval]"=month \
>    -d "recurring[usage_type]"=metered \
>    -d product={{PRODUCT_ID}} \
>    -d tiers_mode=graduated \
>    -d billing_scheme=tiered \
>    -d "expand[0]"=tiers
```

With graduated pricing, 5 projects result in the same charge as volume-based pricing—35 USD total at 7 USD per project. This changes as usage breaks out of the first tier. A customer with more than 5 projects is charged 7 USD per project for the first 5 projects, then 6.50 USD for projects 6 through 10, and finally 6 USD per project thereafter. A customer with 6 projects would be charged 41.50 USD, 35 USD for the first 5 projects and 6.50 USD for the 6th project.

With `tiers_mode=graduated`, the `quantity` is multiplied by the amount that falls into that tier. Then, the total is summed. For example, the total cost for an amount of 6 is 29 USD: 25 USD for the 5 `quantity` that falls into the 1–5 tier, plus 4 USD for the single `quantity` that falls into the 6–10 tier.

QUANTITY AND USAGE AT END OF THE PERIOD	TOTAL FOR GRADUATED TIERED PRICING
1	5 USD
5	25 USD
6	29 USD
20	70 USD
25	75 USD

Adding flat fees

You can specify a flat fee (`flat_amount`) to add to the invoice. This works for both `tiers_mode=volume` and `tiers_mode=graduated`. For example, you can have a flat fee that increases when certain usage thresholds are met:

TIER	AMOUNT (UNIT COST)	FLAT FEE
1-5 (`up_to=5`)	5 USD (`unit_amount=500`)	10 USD (`flat_amount=1000`)
6-10 (`up_to=10`)	4 USD (`unit_amount=400`)	20 USD (`flat_amount=2000`)
10-15 (`up_to=15`)	3 USD (`unit_amount=300`)	30 USD (`flat_amount=3000`)
15-20 (`up_to=20`)	2 USD (`unit_amount=200`)	40 USD (`flat_amount=4000`)
20+ (`up_to=inf`)	1 USD (`unit_amount=100`)	50 USD (`flat_amount=5000`)

In this example, if `quantity` is `12` and `tiers_mode=volume`, the total amount to be billed is `12 * 3 USD + 30 USD = 66 USD`.

If `quantity` is `12` and `tiers_mode=graduated`, the total amount is `5 * 5 USD + 10 USD + 5 * 4 USD + 20 USD + 2 * 3 USD + 30 USD = 111 USD`. A tier can have either a `unit_amount` or a `flat_amount`, or both, but it must have at least one of the two.

If `quantity` is `0`, the total amount is `10 USD` regardless of `tiers_mode`. We always bill the first flat fee tier when `quantity=0`. To bill `0` when there's no usage, set up an `up_to=1` tier with an `unit_amount` equal to the flat fee and omit the `flat_amount`.

Variable pricing

There are two types of variable pricing models:

- **Inline pricing**: You define the price for your customer when you create a subscription, invoice, or Checkout Session.

- **Pay-what-you-want pricing**: The customer fills in the price they pay, like with a tip or donation. This type of variable pricing isn't supported for recurring payments. See Let customers decide what to pay for information about using this type of pricing for single payments.

Inline pricing

In some cases, you might want to use a custom price that hasn't been preconfigured. For example, you might want to use inline prices when you manage your product catalog outside of Stripe.

> **Note**
> You can only create inline prices through the API. Inline prices aren't compatible with Payment Links.

To create an inline price use case, pass in price_data API instead of a price.id API when you create a subscription. For example, to subscribe a customer to a monthly subscription with an inline price:

Command Line — cURL

```
$  curl https://api.stripe.com/v1/subscriptions \
>    -u "sk_test_4eC39HqLyjWDarjtT1zdp7dc:" \
>    -d customer={{CUSTOMER_ID}} \
>    -d "items[0][price_data][unit_amount]"=5000 \
>    -d "items[0][price_data][currency]"=usd \
>    -d "items[0][price_data][product]"={{PRODUCT_ID}} \
>    -d "items[0][price_data][recurring][interval]"=month
```

This creates a monthly recurring price of 50 USD for the basic service offering. By default, prices created with `price_data` are effectively archived (they're marked as `active=false`) so that they you can't reuse them for other customers or subscriptions. You can't update or reuse inline prices after you create them. You can also use `price_data` with these APIs:

- Checkout API
- Invoice Items API
- Subscription Schedules API

Multi-currency prices

A single Price can support multiple currencies. This helps you manage localized pricing when selling internationally.

Create multi-currency Prices

You can create multi-currency Prices in the API or the Dashboard.

Dashboard | API

From the **Product details** page for a product in your Dashboard, click on **+ Add another price** to create a new price. The first currency on your Price will be the default currency. Make sure all your Prices have the same default currency. After selecting the default currency, click on **+ Add more currencies** to add currency options to your Price.

You can search and select from supported currencies. Stripe suggests an exchange rate based on currency values at 12:00 PM EST, but you can pick your own. For currencies that are subject to larger fluctuations, we recommend adding more of a buffer.

After you're done filling in the details, click **Add price** to save it.

Coupons, Promotion Codes, and Shipping Rates also support multi-currency in a similar way to Prices.

Render multi-currency Prices

To show your customer the price in their currency, you can retrieve the multi-currency Price and view its currency_options.<currency>.unit_amount field. The API response won't include `currency_options` by default. To include it in the response, expand the `currency_options` field:

Command Line — cURL

```
$  curl -G https://api.stripe.com/v1/prices/{{PRICE_ID}} \
>    -u "sk_test_4eC39HqLyjWDarjtT1zdp7dc:" \
>    -d "expand[]"=currency_options
```

> **Note**
> To improve latency and avoid problems with rate-limiting, cache the Price instead of re-fetching it every time a customer visits your site.

Use multi-currency Prices

Each purchase uses one of the multi-currency Price's supported currencies, depending on how you use the Price in your integration.

Stripe Checkout | Payment Links | Subscriptions | Quotes | Invoices | Orders