API Reference Guide

Razorpay APIs are completely RESTful and all our responses are returned in JSON.

You can use Razorpay APIs in two modes, `Test` and `Live`. The API key is different for each mode. Know about generating API Keys.

> Integrations
>
> - Looking to integrate your website, ecommerce store or mobile app with Razorpay Payment Gateway? Find the right integration method.
> - Accept payments without a website or app using other Razorpay products, such as Payment Links, Payment Pages, Subscription Links, Invoices and Smart Collect.
> - For S2S integration, contact Support team with your requirements.

API Gateway URL

The Razorpay API Gateway URL is `https://api.razorpay.com/v1`. You need to include this before each API endpoint to make API calls.

> Recommendation
> While sending API requests to Razorpay servers, it is recommended to honor the TTL of the entries and not cache the DNS aggressively at your end. This is applicable when you are not using Razorpay SDKs. However, if you are using Razorpay SDKs, be informed that our SDKs can handle DNS caching and honor the TTLs that are set in the records.

API Authentication

All Razorpay APIs are authenticated using `Basic Auth`. Basic auth requires the following:

- `[YOUR_KEY_ID]`
- `[YOUR_KEY_SECRET]`

Basic auth expects an `Authorization` header for each request in the `Basic base64token` format. Here, `base64token` is a base64 encoded string of `YOUR_KEY_ID:YOUR_KEY_SECRET`.

> Watch Out!
> The `Authorization` header value should strictly adhere to the format mentioned above. Invalid formats will result in authentication failures. Few examples of invalid headers are:
>
> - `BASIC base64token`
> - `basic base64token`
> - `Basic "base64token"`
> - `Basic $base64token`

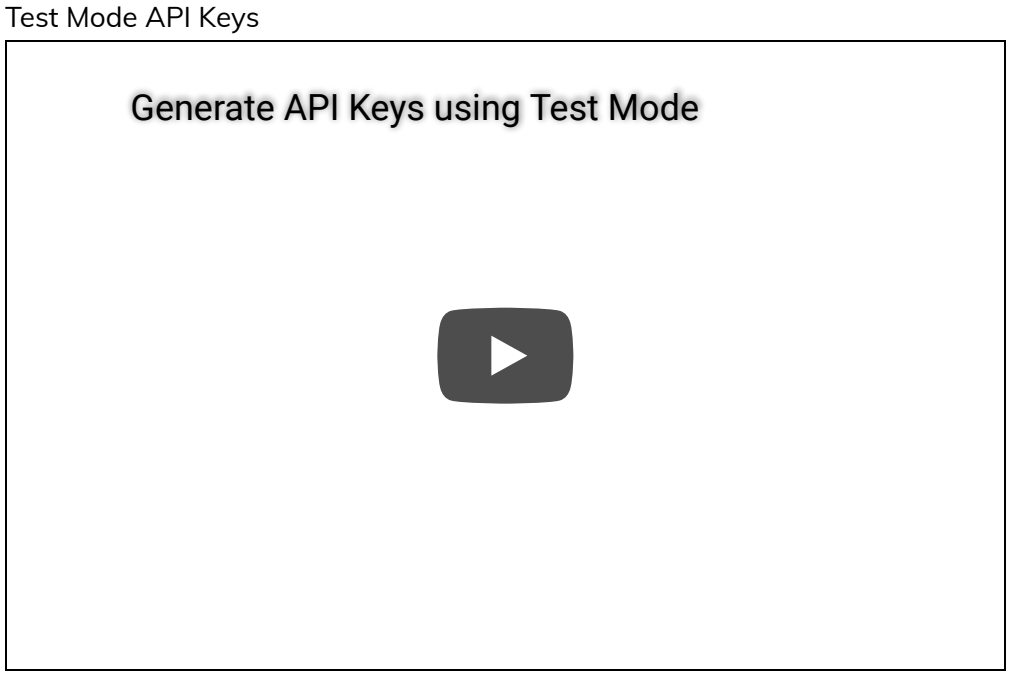
Generate API Key

- Test Mode: The test mode is a simulation mode which you can use to test your integration flow. Your customers will not be able to make payments in this mode.
- Live Mode: When your integration is complete, switch to the live mode and generate live mode API keys. Replace test mode keys with live mode keys in the integration to accept payments from customers.

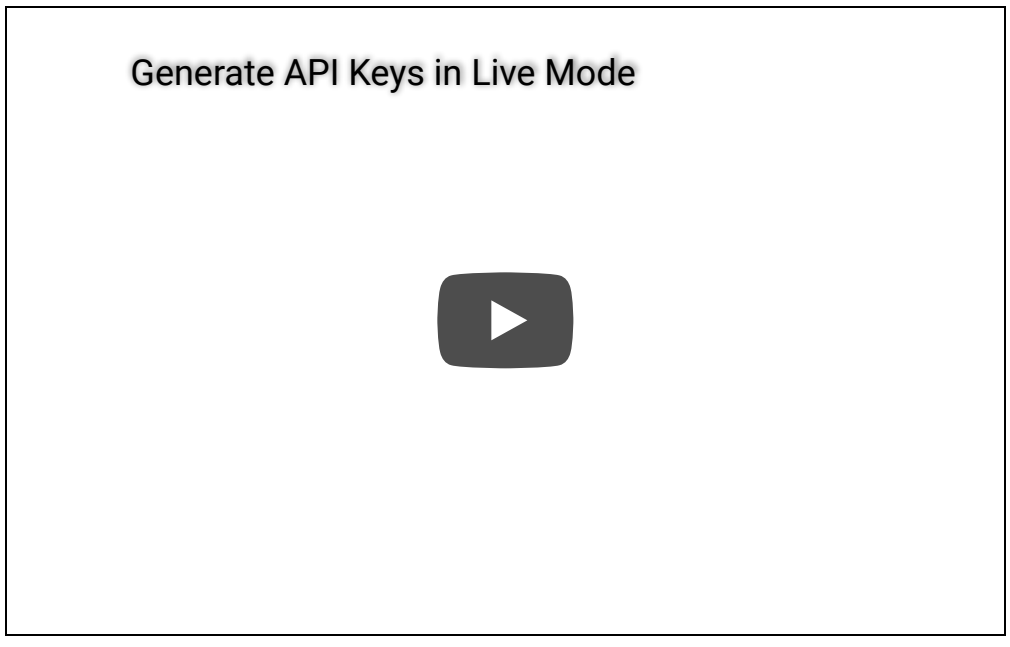
3. Navigate to Settings → API Keys → Generate Key to generate key for the selected mode.

The `Key Id` and `Key Secret` appear on a pop-up page.

Test Mode API Keys



Live Mode API Keys



> Watch Out!
>
> - After generating the keys from the Dashboard, download and save them securely. If you do not remember your API Keys, you need to re-generate it from the Dashboard and replace it wherever required.
> - Do not share your API Key secret with anyone or on any public platforms. This can pose security threats for your Razorpay account .

Errors

All successful responses are returned with HTTP Status code 200. In case of failure, Razorpay API returns a JSON error response with the parameters that detail the reason for the failure.

in exceptions that should be handled in your integration.

Error Response

The error response contains `code` , `description` , `field` , `source` , `step` , `reason` and `metadata` parameters that help you diagnose and solve the error.

Sample Error Response Copy

```
{
  "error": {
    "code": "BAD_REQUEST_ERROR",
    "description": "Authentication failed due to incorrect otp",
    "field": null,
    "source": "customer",
    "step": "payment_authentication",
    "reason": "invalid_otp",
    "metadata": {
      "payment_id": "pay_EDNBKIP31Y4jl8",
      "order_id": "order_DBJKIP31Y4jl8"
    }
  }
}
```

Response Parameters

error	`object`	The error object.
code	`string`	Type of the error.
description	`string`	Description of the error.
field	`string`	Name of the parameter in the API request that caused the error.
source	`string`	The point of failure in the specific operation (payment in this case). For example, customer, business
step	`string`	The stage where the transaction failure occurred. The stages can vary depending on the payment method used to complete the transaction.
reason	`string`	The exact error reason. It can be handled programmatically.
metadata ⌄	`object`	Contains additional information about the request.

Know more about Error Codes.

Entity

Every API response contains entities that are shared across different endpoints. There are some common attributes for every entity.

entity	`string`	Indicates the type of the entity.
id	`integer`	A unique identifier of the entity.

Example Order Entity

Copy

```
{
  "id": "order_6JUYuvmgCLfgjY",
  "entity": "order",
  "amount": 50000,
  "currency": "INR",
  "attempts": 0,
  "status": "created",
  "receipt": "receipt#42",
  "notes": [],
  "created_at": 1474013013
}
```

In an entity, the attributes can be utilized to make entity-specific API calls. For example, you can fetch the payment ID from an `order.paid` webhook and use it to initiate a refund for that payment.

Collection Entity

Razorpay API also supports returning multiple entities for a single request. This response also has another entity `collection` . For the `collection entity` , the following parameters are common.

entity	`string`	Indicates the type of the entity. For example, `collection` .
count	`integer`	Indicates the number of `items` are returned. For example, `2` .
items	`array`	The list of entities.

Sample Collection Response

Copy

```
{
  "count":2,
  "entity":"collection",
  "items":[
    {
      "id":"pay_29QQoUBi66xm2f",
      "entity":"payment",
      "amount":500,
      "currency":"INR",
      "status":"created",
      "amount_refunded":0,
      "refund_status":null,
      "email":"gaurav.kumar@example.com",
      "contact":"9364591752",
      "error_code":null,
```

```
    },
    {
      "id":"pay_19btGlBig6xZ2f",
      "entity":"payment",
      "amount":500,
      "currency":"INR",
      "status":"created",
      "amount_refunded":0,
      "refund_status":null,
      "email":"gaurav.kumar@example.com",
      "contact":"9364591752",
      "error_code":null,
      "error_description":null,
      "notes":[],
      "created_at":1400826750
    }
  ]
}
```

Notes

The majority of the entities allow `notes` object to store additional information and preserve data that is relevant to your integration. It is not used by Razorpay for any operational purposes.

The `notes` object is a set of key-value pairs that can be used to store additional information about the entity. It can hold a maximum of 15 key-value pairs, each 256 characters long (maximum).

Example Order Entity Copy

```
{
  "id": "order_6JUYuvmgCLfgjY",
  "entity": "order",
  "amount": 50000,
  "currency": "INR",
  "attempts": 0,
  "status": "created",
  "receipt": "receipt#42",
  "notes": {
    "my_store_id": "ref#123123123",
    "my_user_id": "user_0316"
  },
  "created_at": 1474013013
}
```

For example, you can store the notes related to:

- Billing or shipping address of the initiated payment.
- Reference ID generated for an order.

Query Parameters

You can send the following query parameters to apply filters on the bulk data and view only the appropriate response.

from	`integer`	Timestamp, in Unix, after which the entities are created.
to	`integer`	Timestamp, in Unix, before which the entities are created.

	pagination, in combination with `skip` .
skip	`integer` Number of entities to skip. Default is 0. This can be used for pagination, in combination with `count` .

Rate Limits

Razorpay employs a request rate limiter that limits the number of requests received by the API within a time window. This is to maintain system stability in the event of unintentional high traffic loads.

While integrating with any APIs, watch for `HTTP status code 429` and build the retry mechanism based on the requirement.

To make the best use of the limits, it is recommended to use an Exponential backoff/stepped backoff strategy to reduce request volume and stay within the limit. It is also recommended to have some randomization within the backoff schedule to avoid the thundering herd effect.