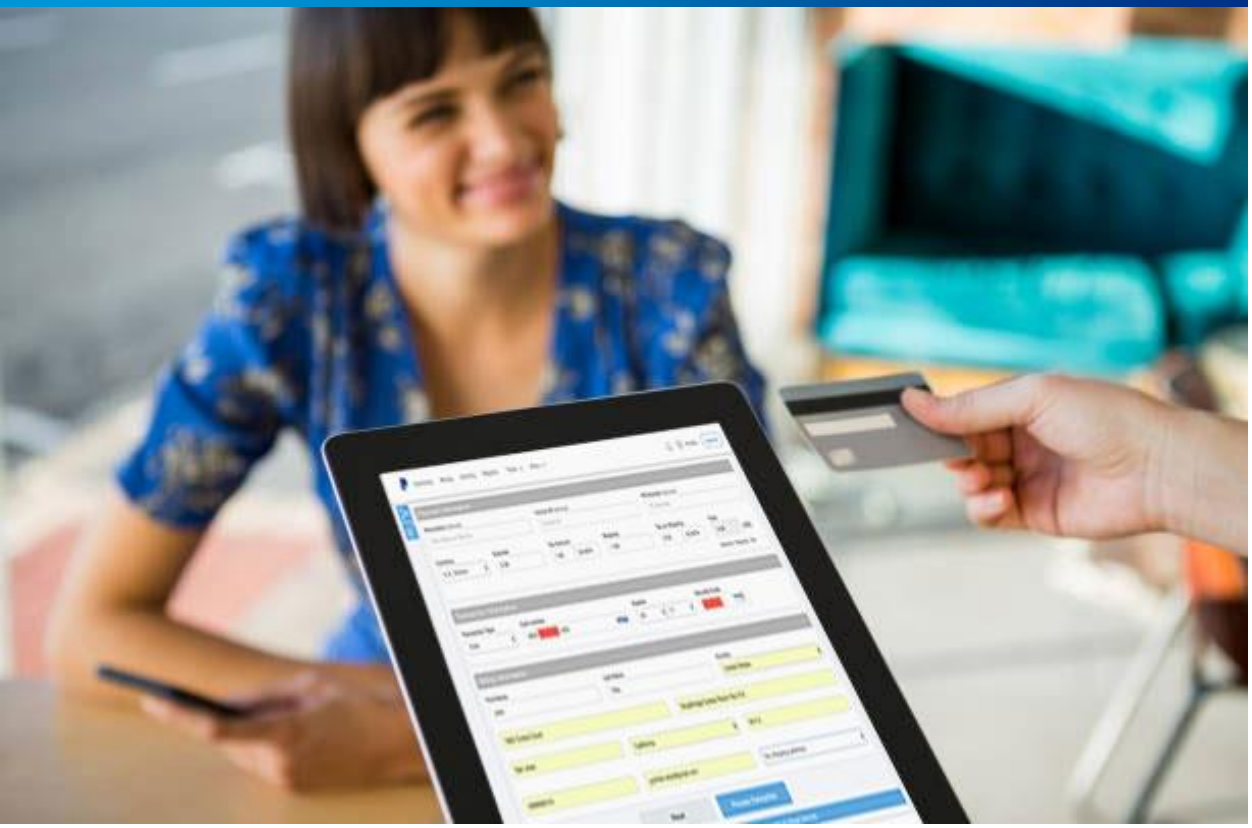
Integrating PayPal Checkout in India

INTEGRATION GUIDE



Contents

PayPal ©2020 PayPal.

Copyright Information

© 2020 PayPal.

PayPal is a registered trademark of PayPal, Inc. The PayPal logo is a trademark of PayPal, Inc. Other trademarks and brands are the property of their respective owners. The information in this document belongs to PayPal, Inc. It may not be used, reproduced, or disclosed without the written approval of PayPal, Inc.

Consumer advisory: The PayPal™ payment service is regarded as a stored value facility under Singapore law. As such, it does not require the approval of the Monetary Authority of Singapore. You are advised to read the terms and conditions carefully.

Notice of Non-Liability

PayPal, Inc. is providing the information in this document to you “AS-IS” with all faults. PayPal, Inc. makes no warranties of any kind (whether express, implied or statutory) with respect to the information contained herein. PayPal, Inc. assumes no liability for damages (whether direct or indirect), caused by errors or omissions, or resulting from the use of this document or the information contained in this document or resulting from the application or use of the product or service described herein. PayPal, Inc. reserves the right to make changes to any information herein without further notice.

1. Prerequisites

To complete an integration, you will need:

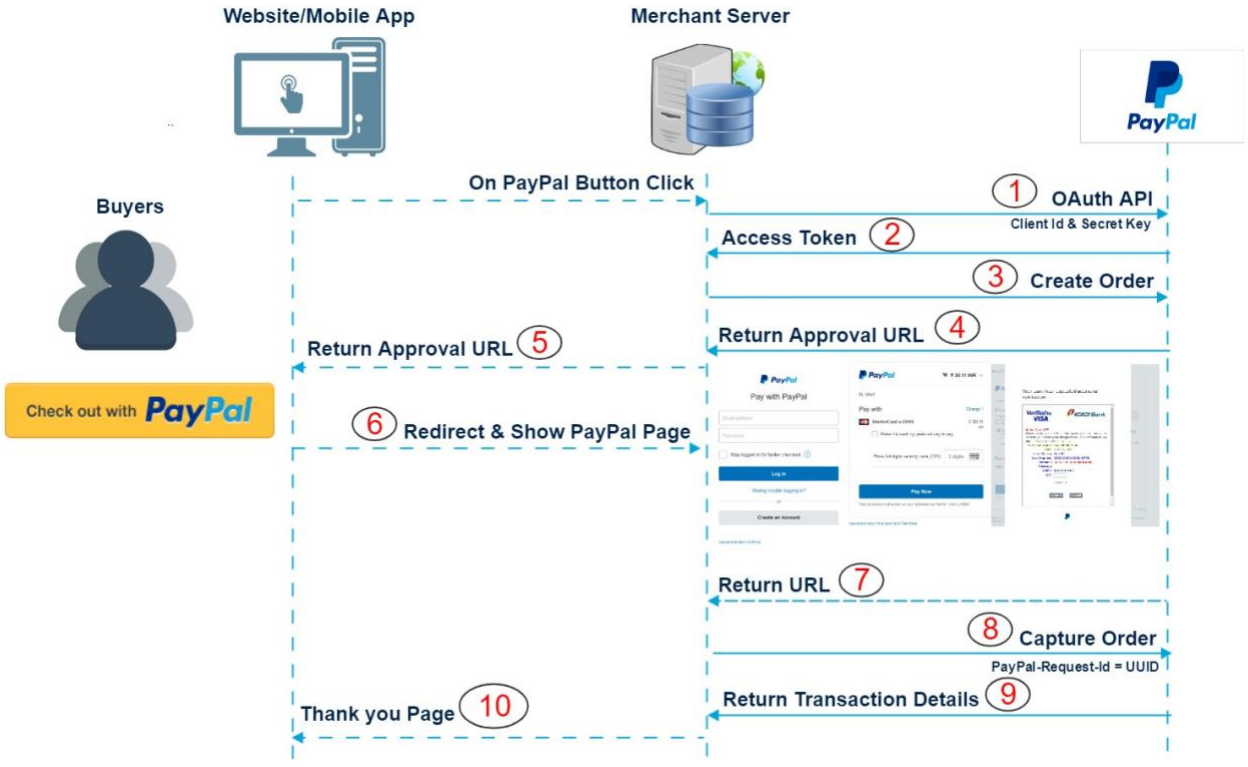
- Basic development knowledge of REST APIs.
- A PayPal Business account.

2. Overview

This document provides guidance for developers in India to integrate PayPal Checkout.

2.1 Quick Steps

The following image and callouts provide a high-level overview of how PayPal Checkout works.



1. When the PayPal button is clicked, control goes to the website/app server and an OAuth API call is initiated.
2. On success, an access token is posted back in response. Use this access token for all further API calls.

NOTE: An access token is valid for 8 hours.

3. Initiate the Create Order API call with all recommended checkout details (for example, the amount, buyer details, shipping address, etc.)
4. On success, an approval URL is posted back in the response to the web/app server.

5. Redirect the buyer to the approval URL, either from the website or mobile app.
6. The buyer is landed on the PayPal page and can complete payment either through the PayPal login or the credit/debit card option.
7. After successful payment authorization, control is redirected back to the Return URL of the web/app server.
8. Initiate the Capture Order API call by posting dynamic Order ID to complete the transaction.
9. On success, the order amount is deducted from the buyer's account, and complete transaction details are posted through the response payload. Check that the transaction status is Completed, and store the Transaction ID for future reference.
10. Redirect the buyer to the website "Thank You" page if the transaction status is successful.

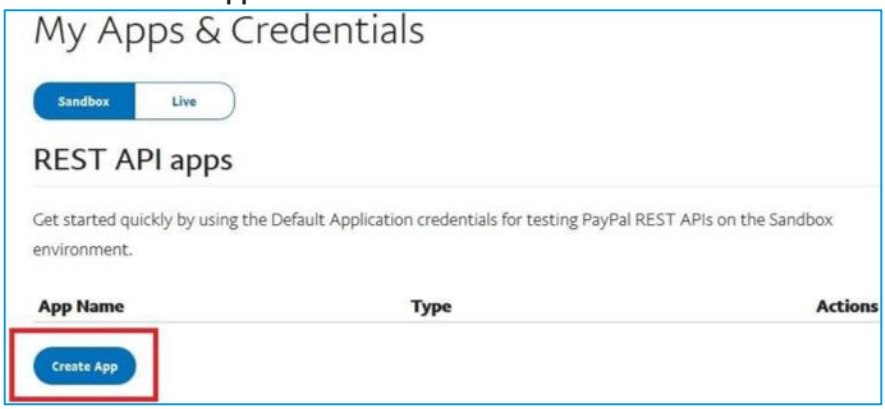
3. Integration Steps

Complete the following steps to integrate PayPal Checkout.

3.1 Create a REST API app

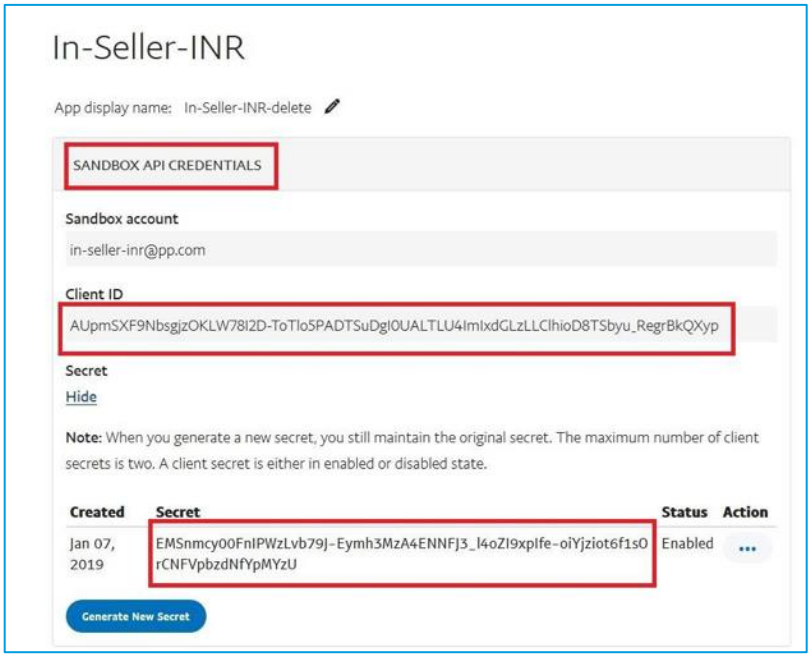
Create a REST API app to generate API credentials.

1. Click **Log into Dashboard** at https://developer.paypal.com and log in with your PayPal account credentials.
2. Click **My Apps & Credentials** on the left side of the page, toggle to the **Sandbox** tab, and click **Create App**.



3. Enter an **App Name** and click **Create App**.

4. In the app details page that opens, copy the client ID and secret for the sandbox environment.



5. To get live credentials, toggle to the **Live** tab and repeat these steps.

3.2 Make Server-Side API Calls

Complete the following server-side API calls.

These are the base URLs:

Sandbox	https://api.sandbox.paypal.com
Live	https://api.paypal.com

3.2.1 Get Access Token API

Create a unique access token through the OAuth API to make subsequent payment API calls.

POST	{{BASE_URL}}/v1/oauth2/token

	Property	**Value**
Headers	Accept	application/json
	Accept-Language	en_US
	Username	<client_id>
	Password	<secret_key>

Sample Request Body

```
grant_type = client_credentials
```

Sample Response Body

```
{
"scope": "https://uri.paypal.com/services/subscriptions
https://api.paypal.com/v1/payments/.*
https://api.paypal.com/v1/vault/credit-card
https://uri.paypal.com/services/applications/webhooks openid
https://uri.paypal.com/payments/payouts
https://api.paypal.com/v1/vault/credit-card/.*",
"nonce": "2019-02-08T18:30:28ZCl54Q_OlDqP6-
4D03sDT8wRiHjKrYlb5EH7Di0gRrds",
"access_token": "<Access-Token>", // Pass this token in headers of all
subsequent API calls (Valid 8 hrs)
"token_type": "Bearer",
"app_id": "APP-80W284485P519543T",
"expires_in": 32398
}
```

NOTE: For more details, see https://developer.paypal.com/docs/api/overview/#get-an-access-token.

3.2.2 Create Order API

Use the Create Order API to create a payment.

POST	{{BASE_URL}}/paypal.com/v2/checkout/orders

	Property	Value
Headers	Content-Type	Application/json
	Authorization	Bearer {{access_token}}

Sample Request Body

```
{
    "intent": "CAPTURE", //Capture payment immediately after the customer
makes a payment
    "application_context": {
        "brand_name": "MY BRAND", //The label that overrides the business
name in the PayPal account on the PayPal site
        "locale": "en-IN",
        "user_action": "PAY_NOW", //Process the payment immediately when
the customer clicks Pay Now
        "return_url": "http://localhost/return", //Payment successful
redirection URL
        "cancel_url": "http://localhost/cancel", //Payment cancel
redirection URL
        "payment_method": {
            "payer_selected": "PAYPAL"
        }
    },
    "payer": { //Pass and prefill buyer details like first name, last
name, email address and phone number
        "name": {
            "given_name": "John",
            "surname": "Doe"
        },
        "email_address": "customer@example.com",
        "phone": {
            "phone_number": {
                "national_number": "9874563210"
            }
        },
        "address": { //Pass and prefill buyer billing address details
            "address_line_1": "10, east street",
            "address_line_2": "second building",
```

```
            "admin_area_2": "Mumbai",
            "admin_area_1": "Maharashtra",
            "postal_code": "400029",
            "country_code": "IN"
        }
    },
"purchase_units": [{
        "amount": {
            "currency_code": "INR",
            "value": "170.00", //Total amount
            "breakdown": {
                "item_total": {
                    "currency_code": "INR", //Three-character ISO-4217
currency code that identifies the currency
                    "value": "180.00"
                },
                "discount": { //The discount for all items within a given
purchase_unit
                    "currency_code": "INR",
                    "value": "10.00"
                }
            }
        },
        "items": [ //Pass line item details
            {
                "name": "T-Shirt",
                "description": "Green XL",
                "sku": "sku01",
                "unit_amount": {
                    "currency_code": "INR",
                    "value": "90.00"
                },
                "quantity": "1",
                "category": "PHYSICAL_GOODS" //Item category type
            },
            {
                "name": "Shoes",
                "description": "Running, Size 10.5",
                "sku": "sku02",
                "unit_amount": {
                    "currency_code": "INR",
                    "value": "45.00"
                },
                "quantity": "2",
                "category": "PHYSICAL_GOODS"
            }
        ],
        "soft_descriptor": "CC_STATEMENT_NAME", //Payment descriptor on
account transactions on the customer's credit card statement
        "invoice_id": "INV-1234567890", //Pass the invoice ID. It needs to
be unique. It will help to identify the transaction in PayPal side
        "shipping": { //Pass and prefill shipping address details
            "name": {
                "full_name": "John Doe"
            },
            "address": { //Pass and prefill buyer shipping address details
                "address_line_1": "10, east street",
                "address_line_2": "first building",
                "admin_area_2": "Mumbai",
                "admin_area_1": "Maharashtra",
                "postal_code": "400029",
                "country_code": "IN"
            }
        }
```

Sample Response Body

```
{
"id": "1HX35072P9443720D", //This is the Order ID that you need to use in
the capture call
    "intent": "CAPTURE",
    "purchase_units": [
        {….  },
        "address": {…. },
    "links": [
        {…..  },
        {
            "href":
"https://www.paypal.com/checkoutnow?token=1HX35072P9443720D", //PayPal
approval url at where you need to redirect user to make the payment
            "rel": "approve",
            "method": "GET"
        },….
```

3.2.3 Capture Order API

Use the Capture Order API to execute the payment.

NOTE: Pass the order ID dynamically.

POST	{{BASE_URL}}/v2/checkout/orders/{{order_id}}/capture

	Property	Value
Headers	Content-Type	Application/json
	Authorization	Bearer {{access_token}}
	PayPal-Request-Id	Unique user-generated ID

Sample Request Body

No parameters required.

Sample Response Body

```
{
   "id":"5O190127TN364715T", //Order ID. Save the ID in DB for future
reference
   "status":"COMPLETED",
   "payer{…. },"purchase_units":[ //Save the Payer ID in DB for future
reference
      {
         ….
         "shipping":{ …. }
         },
         "payments":{
            "captures":[
               {
                  "id":"3C679366HH908993F", //Save the ID in DB for future
reference
                  "status":"COMPLETED", //This status is used to confirm
that the payment is properly completed
                  "amount":{ ….     },
                  "seller_protection":{ ….  },   ….  },
```

3.2.4 Show Order Details API

Shows details for an order, by ID.

NOTE: Pass the order ID dynamically.

POST	{{BASE_URL}}/v2/checkout/orders/{order_id}

	Property	Value
Headers	Content-Type	Application/json
	Authorization	Bearer {{access_token}}

Sample Request Body

No parameters required.

Sample Response Body

```
{
  "id": "5O190127TN364715T",
  "status": "CREATED",
  "intent": "CAPTURE",
  "purchase_units": [
    {
      "reference_id": "d9f80740-38f0-11e8-b467-0ed5f89f718b",
      "amount": {
        "currency_code": "USD",
        "value": "100.00"
      }
    }
  ],
  "create_time": "2018-04-01T21:18:49Z",
  "links": [
    {
      "href":
"https://api.paypal.com/v2/checkout/orders/5O190127TN364715T",
      "rel": "self",
      "method": "GET"
    },
    {
      "href":
"https://api.paypal.com/checkoutnow?token=5O190127TN364715T",
      "rel": "approve",
      "method": "GET"
    },
    {
      "href":
"https://api.paypal.com/v2/checkout/orders/5O190127TN364715T/capture",
      "rel": "capture",
      "method": "POST"
    }
  ]
```

Post-Payment Check

For future reference, make sure that you save the invoice ID and transaction order ID for the transaction in your database.

3.2.5 Refund Order API

Use the Refund Order API to refund the payment.

POST	{{BASE_URL}}/paypal.com/v2/payments/captures/{{capture_id}}/refund

	Property	Value
Headers	Content-Type	Application/json
	Authorization	Bearer {{access_token}}

Sample Request Body

```
"amount": {
 "value": "10.00",
 "currency_code": "USD",
 },
 "invoice_id": "INVOICE-123",
 "note": "Defective Product"
 }
```

Sample Response Body

```
{
  "id": "1JU08902781691411",
  "status": "COMPLETED",
  "links": [
    {
      "rel": "self",
      "method": "GET",
      "href":
"https://api.paypal.com/v2/payments/refunds/1JU08902781691411"
    },
    {
      "rel": "up",
      "method": "GET",
      "href":
"https://api.paypal.com/v2/payments/captures/2GG279541U471931P"
    }
  ]
}
```

4. Error Handling

Refer to the following topic for common PayPal RESTful error codes and how to handle them: https://developer.paypal.com/docs/api/reference/api-responses/

The following list contains detail about specific error conditions:

- **Validation Error (HTTP Status Code 400)** – Caused by one or more fields containing erroneous data.
- **For 500 (Internal Server Error) and 503 (Service Unavailable)** – If you come across these errors capturing the payment, try your request again (re-trigger the payment capture).

PayPal returns more information about the error in the body of the response.
The following parameters are included:

- **name** – The name of the error.
- **message** – A description of the error.
- **information link** – A link to further information on the error, if available.
- **details** – Additional details about the error, if available.

Sample Error Response

```
{
"name": "INSTRUMENT_DECLINED",
"details": [],
"message": "The instrument presented was either declined by the processor
or bank, or it can't be used for this payment.",
"information_link":
"https://developer.paypal.com/docs/api/payments/#errors",
"debug_id": "e9d282fbf2d70"
}
```

Sample Error Response with Details

```
{
"name": "VALIDATION_ERROR",
"details": [{
"field": "zip",
"issue": "Value is invalid"
}],
"message": "Invalid request - see details",
"information_link":
"https://developer.paypal.com/docs/api/payments/#errors",
"debug_id": "e6dfc3766c374"
}
```

There are additional errors that PayPal might return in a response, depending on business case. Please refer to the following topic for more information about possible orders-related errors:

https://developer.paypal.com/docs/api/orders/v2/#errors

We recommend you read the following error-related topics for any integration:

- **Handle Errors** - https://developer.paypal.com/docs/checkout/integration-features/handle-errors/
- **Handle Funding Failures** - https://developer.paypal.com/docs/checkout/integration-features/funding-failure/

For example, if (error== INSTRUMENT_DECLINED) {retry for Payment}

- **Show a Cancellation Page** – https://developer.paypal.com/docs/checkout/integration-features/cancellation-page/
- **Troubleshooting Integration Queries** – https://developer.paypal.com/docs/checkout/troubleshoot/

5. Webhooks

The PayPal REST APIs use webhooks for event notification.

Webhooks are HTTP callbacks that receive notification messages for events. To create a webhook at PayPal, users configure a webhook listener and subscribe it to events. A webhook listener is a server that listens at a specific URL for incoming HTTP POST notification messages that are triggered when events occur. PayPal signs each notification message that it delivers to your webhook listener.

Refer to the following links for more details about webhooks:

- **Webhooks Overview** – https://developer.paypal.com/docs/integration/direct/webhooks/rest-webhooks/
- **Webhooks Integration Steps** – https://developer.paypal.com/docs/integration/direct/webhooks/rest-webhooks/#integration-steps
- **Adding Webhooks Events** – https://developer.paypal.com/docs/checkout/integration-features/add-webhooks/

6. Testing

6.1 Sandbox Testing

1. Create sandbox buyer and business accounts in the Developer Dashboard. Click Log into Dashboard at https://developer.paypal.com and create these accounts on the **Sandbox>Accounts** page. Use these accounts to verify all points mentioned in the checklist and check the entire payment flow in the sandbox environment.
2. Log in to https://www.sandbox.paypal.com with your sandbox business account to confirm that the funds have been received (minus any processing fees).
3. Log in to https://www.sandbox.paypal.com with your sandbox buyer account to confirm that the funds have been sent.

6.2 Live Testing

1. Update all API endpoints to api.paypal.com in the staging/test server and use your LIVE client ID & secret to test PayPal LIVE transactions in the staging environment.
2. Verify your live transactions from both the merchant's and buyer's perspective and verify all points mentioned in the checklist.
3. Log in to your PayPal account using your real PayPal business account to confirm that the funds have been received (minus any processing fees).

4. Log in to your PayPal account using a real PayPal buyer account to confirm that the funds have been sent.
5. Verify *partial* and *full* refund types are supported as appropriate and webhooks event payloads are being received correctly.

7. Checklist

PayPal Acceptance Mark & Buyer Experience

- Pass the billing address, shipping address, and buyer information like first name, last name, mobile number and email address details in the API call to prepopulate the same in the PayPal checkout pages.
- Append the country code and locale code in the approval/redirect URL (only for INR payments). Example:

```
https://www.paypal.com/checkoutnow?token=11A11111A1111111A&locale.x=en_IN&
country.x=IN
```

- The **Pay Now** button should be displayed on the PayPal review page.
- Check that the correct capitalization of “PayPal” is always used in text and images (instead of “Paypal”, “paypal”, “Pay Pal”, etc.).
- Place the PayPal logo and/or PayPal acceptance mark on your website. You can find PayPal logos and value props at the Logo Center (https://www.paypal.com/in/webapps/mpp/logo-center). The PayPal logo should have parity with other logos.
- Store the PayPal transaction ID (Capture Order), order ID (Create Order), and payer ID (Redirect URL) for future reference.

Payment Flow

- Check whether the contents of the buyer’s shopping cart are passed using appropriate line item detail parameters (name, description, quantity, price, etc.).
- Check if the discount amount is passed as negative value under discounts.
- Make sure the invoice ID is passed. Pass you own order reference as invoice ID.
- Make sure no additional surcharge or fee is added for PayPal transactions.
- Make sure the value of the payment status response is checked and pending payments are handled appropriately.
- Verify forward and reverse money movement and ensure partial and full refund types are supported as appropriate.

API and Error Handling

- **Handling insufficient funds error** – If the buyer’s selected funding source fails with an instrument declined error/insufficient funds error, make sure the buyer can choose from another funding source.

Mobile Integration

- Ensure PayPal pages open directly in the custom Chrome tab for Android or Safari view controller for iOS based on the device.

8. Reference Links

- Smart Payment Button Overview: https://developer.paypal.com/docs/checkout/
- Basic Integration Methods: https://developer.paypal.com/docs/checkout/integrate/
- REST API Order V2 API Reference: https://developer.paypal.com/docs/api/orders/v2/
- PayPal Checkout Integration Best Practices: https://developer.paypal.com/docs/checkout/best-practices/

9. Sign-off from Integration Engineer

Proper approval is required from a PayPal integration engineer before going live.

10. Go Live

1. Replace your sandbox credentials with live credentials (client ID, secret). Example:

```
<script
  src="https://www.paypal.com/sdk/js?client-id=<<SB_CLIENT_ID>>">
</script>
```

2. Change all API endpoints from https://api.sandbox.paypal.com to https://api.paypal.com, and use your live client ID and secret for these calls.
3. Verify live transactions from both the merchant's and buyer's perspective:
    - Log in to your PayPal account using your real PayPal business account to confirm that the funds have been received (minus any processing fees).
    - Log in to your PayPal account using a real PayPal buyer account to confirm that the funds have been sent.

NOTE: Please refer to this link for more details: https://developer.paypal.com/docs/checkout/integrate/#8-go-live