PayPal API Flows

Standard Checkout flow

1. Create Order with intent = "capture": https://api-m.sandbox.paypal.com/v2/checkout/orders
2. It's a best practice to add line-item details (such as the description and item name within the items array in the purchase_unit). Shipping address information can also be passed in the purchase_unit.
3. Redirect browser to the "payer_action" HATEOAS link from the API response.
    a. Note: Depending on what is passed in the Create Order request, the HATEOAS link may be returned as "approve"
4. Log into PayPal using the buyer's PayPal account and approve the sale. You can also set a shipping address if necessary.
5. Redirect back to the merchant server (if not using the JavaScript SDK for the client, you will need to pass in the return_url in the original Create Order request).
6. Capture Order: {{host}}/v2/checkout/orders/{{order_id}}/capture
7. Get Order Details: {{host}}/v2/checkout/orders/{{order_id}}
    a. Use this API call to retrieve the information about the Order. This can be called any time after the Create Order was made.

Authorize and Capture Flow

1. Create Order with intent = "authorize"
2. Redirect browser to the "payer_action" HATEOAS link from the API response.

    a. Note: Depending on what is passed in the Create Order request, the HATEOAS link may be returned as "approve"

3. Log into PayPal using the buyer's PayPal account and approve the sale. You can also set a shipping address if necessary.
4. Redirect back to the merchant server (if not using the JavaScript SDK for the client, you will need to pass in the return_url in the original Create Order request).
5. Authorize Order: {{host}}/v2/checkout/orders/{{order_id}}/authorize
    a. This returns an authorization ID in the response. Take note of this as the authorization ID is what is used to capture the funds (move the money from the buyer to the seller).
6. Capture Authorization: {{host}}/v2/payments/authorizations/{{authorization_id}}/capture
    a. You can pass in an amount in this request if you wish to do a partial capture. Otherwise, don't pass in an amount to capture the full amount that was set in the Create Order.
7. You can void the authorization after the Auth ID has been created. However, the void must be done before you make the Capture Authorization call: https://api-m.sandbox.paypal.com/v2/payments/authorizations/{{authorization_id}}/void
8. We could create a new Authorization ID within 29 days of creating the initial authorization. This is used to restart the honor period (time when the funds are held) prior to the capture. However, the reauthorization can only be called 3 days after the Authorization ID was created: {{host}}/v2/payments/authorizations/{{authorization_id}}/reauthorize

Refund Flow

1. After the transaction has been completed (the Capture call has been made and successfully completed), you can refund money

back to the buyer's account. You can pass in an amount to this call if you wish to do only a partial refund.

2. Refund: {{host}}/v2/payments/captures/{{capture_id}}/refund

Patch Order Flow

1. After the Create Order, but before either the Authorize Order or Capture Order APIs are called, you can update certain information within the Order.
2. Call the Update Order: PATCH {{host}}/v2/checkout/orders/{{order_id}}
3. Refer to the code sample from the Postman collection to see an example of how the information is passed.

Save Payment Method with Purchase Flow with JavaScript SDK

1. Generate User ID Token: https://api-m.sandbox.paypal.com/v1/oauth2/token
    a. Pass in response_type=id_token in the request
    b. For a returning buyer that already has a saved payment method in the vault, you'll also need to pass in target_customer_id with the customer ID created by PayPal at the time of the original Save Payment Method.
    c. Pass the id_token returned in this response to the JS SDK using the data-user-id-token attribute.
2. Create Order with PayPal Payment Source: https://api-m.sandbox.paypal.com/v2/checkout/orders
3. Authorize or Capture the Order
    a. Capture Order: {{host}}/v2/checkout/orders/{{order_id}}
    b. Authorize Order: {{host}}/v2/checkout/orders/{{order_id}}/authorize
4. Store the Vault ID and the Customer ID for future purchases.

5. Refer to this page for more details about the vault flow: https://developer.paypal.com/docs/checkout/save-payment-methods/during-purchase/js-sdk/paypal/

Save Payment Method without Purchase Flow with JavaScript SDK

1. Generate User ID Token: https://api-m.sandbox.paypal.com/v1/oauth2/token
    a. Pass in response_type=id_token in the request
    b. For a returning buyer that already has a saved payment method in the vault, you'll also need to pass in target_customer_id with the customer ID created by PayPal at the time of the original Save Payment Method.
    c. Pass the id_token returned in this response to the JS SDK using the data-user-id-token attribute.
2. Create Setup Token: https://api-m.sandbox.paypal.com/v3/vault/setup-tokens
3. Redirect the buyer to the "approve" HATEOAS link.
4. Log into the PayPal account as the buyer. Click the continue button to approve the saving the payment method.
5. Create Payment Method Token: https://api-m.sandbox.paypal.com/v3/vault/payment-tokens
    a. There's currently an issue where the payment_source > token > type does not include the required value "SETUP_TOKEN" in the SDK due to a visibility issue with the API schemas. This is being corrected. But, as a workaround, you can hard code the SETUP_TOKEN type instead.
6. Refer to this page for more details about the vault flow: https://developer.paypal.com/docs/checkout/save-payment-methods/during-purchase/js-sdk/paypal/

Purchase Using Saved Payment Method (Returning Buyer)

1. Create Order with intent = “capture” or intent = “authorize”: https://api-m.sandbox.paypal.com/v2/checkout/orders
2. In the request, be sure to add the saved Vault ID from previous steps into the payment_source.paypal.vault_id or payment_source.card.vault_id

Purchase Using 3D Secure Authentication (Single Step)

1. Create Order with intent = “capture”: https://api-m.sandbox.paypal.com/v2/checkout/orders
2. In the request, be sure to add the payment_source > card object. Within the card object you can pass in a credit card number to simulate different 3D Secure responses.
3. You will also need to pass in the “attributes” object with the “verification” set to “SCA_ALWAYS” to trigger the 3D Authentication flow with each request. You can also use “SCA_WHEN_REQUIRED”, but this may not trigger the 3D Authentication flow for all the test credit card numbers.
4. In API response there should be a payer-action link. You will need to redirect to this link to authenticate the credit card. In some cases, a pop up to the simulator page will appear asking for an OTP. The OTP will usually be printed on the web page itself. In some other cases, the link will just redirect back to the return URL specified in the experience_context, but this means that the authentication was already completed in the background.
4. Get Order Details: {{host}}/v2/checkout/orders/{{order_id}}
    a. Use this API call to retrieve the information about the Order. Within the response should be the payment_source > card > authentication_result object with the details about the 3D Secure Authentication.
5. Capture Order: {{host}}/v2/checkout/orders/{{order_id}}/capture

6. Refer to this page for more details about the 3D Secure Authentication flow: https://developer.paypal.com/docs/checkout/advanced/customize/3d-secure/api/

Purchase Using 3D Secure Authentication (Multi Step)

5. Create Order with intent = “capture”: https://api-m.sandbox.paypal.com/v2/checkout/orders
6. In the request, do not add the payment_source > card object.
7. Once you have run the API request, call the Capture Order API: {{host}}/v2/checkout/orders/{{order_id}}/capture
8. In the Capture Order request, be sure to add the payment_source > card object. Within the card object you can pass in a credit card number to simulate different 3D Secure responses.
9. You will also need to pass in the “attributes” object with the “verification” set to “SCA_ALWAYS” to trigger the 3D Authentication flow with each request. You can also use “SCA_WHEN_REQUIRED”, but this may not trigger the 3D Authentication flow for all the test credit card numbers.
10. In API response there should be a payer-action link. You will need to redirect to this link to authenticate the credit card. In some cases, a pop up to the simulator page will appear asking for an OTP. The OTP will usually be printed on the web page itself. In some other cases, the link will just redirect back to the return URL specified in the experience_context, but this means that the authentication was already completed in the background.
11. Get Order Details: {{host}}/v2/checkout/orders/{{order_id}}
12. Use this API call to retrieve the information about the Order. Within the response should be the payment_source > card >

authentication_result object with the details about the 3D Secure Authentication.

13. Use this API call to retrieve the information about the Order. Within the response should be the payment_source > card > authentication_result object with the details about the 3D Secure Authentication.
14. Capture Order: {{host}}/v2/checkout/orders/{{order_id}}/capture
15. This may return a "Transaction Refused" error message in our Sandbox environment.
16. Calling the Get Order Details again will show the captures object with the capture ID and information about the completed transaction.
17. Refer to this page for more details about the 3D Secure Authentication flow: https://developer.paypal.com/docs/checkout/advanced/customize/3d-secure/api/