Customer Disputes API

1. Introduction

In this manual, you will learn to integrate the Dispute API that helps PayPal merchants, partners, and external developers to list disputes, provide evidence, accept claims, show dispute details, and appeal disputes. An Application Programming Interface (API) is the term for communication methods that work between the software components. When it is used in the context of web development, an API is defined as a set of specifications with the HTTP request message protocol, and the structure of the replies, which is usually XML or a JSON format.

A dispute is made when a customer files a case with PayPal or to asks to his/her bank (or credit card company) to do a chargeback. When a customer disputes your charge, you can provide evidence to show that the charge is legitimate. To provide new evidence or appeal a dispute, you submit a proof of a delivery or proof of refund document or a receipt, which can include logs.

To use the Customer Disputes API, you must:

- Be a merchant or a partner in the PayPal Partner Program.
- Create a PayPal app and get an *access token*. When you create a PayPal app, PayPal generates a set of OAuth `client_id` and secret keys for the application. PayPal generates these keys for both the Sandbox and Live environments. To get an *access token*, pass the `client-id:secret` credentials in the `Authorization` header. You use the access token for authentication when you make REST API requests.

This guide serves as a reference for getting started with the Customer Disputes integration. If you need additional information that is not enclosed in this guide, send an email message to the PayPal Professional Service team. It assumes that the readers have the required expertise for such an integration.

2. Customer Disputes flows

In figure 1, you can see that the flow follows the industry standard for disputes, is scalable, generic and can be automated.

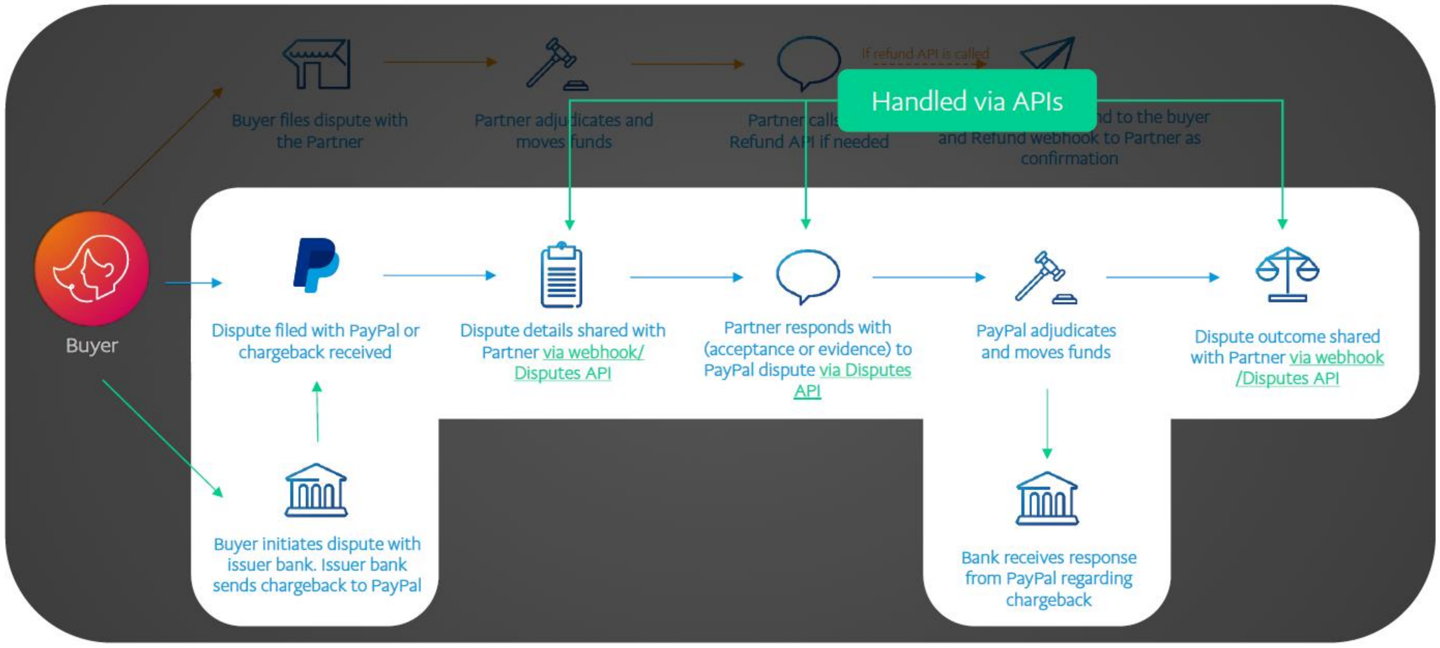

Buyer files dispute with the Partner
Partner adjudicates and moves funds
Partner calls the Refund API if needed
If refund API is called
and Refund webhook to Partner as confirmation
Handled via APIs
Buyer
Dispute filed with PayPal or chargeback received
Dispute details shared with Partner via webhook/ Disputes API
Partner responds with (acceptance or evidence) to PayPal dispute via Disputes API
PayPal adjudicates and moves funds
Dispute outcome shared with Partner via webhook /Disputes API
Buyer initiates dispute with issuer bank. Issuer bank sends chargeback to PayPal
Bank receives response from PayPal regarding chargeback

Figure 1. Disputes API flow

Merchants can also handle disputes that come directly to them in a customized way, as you can see in figure 2.

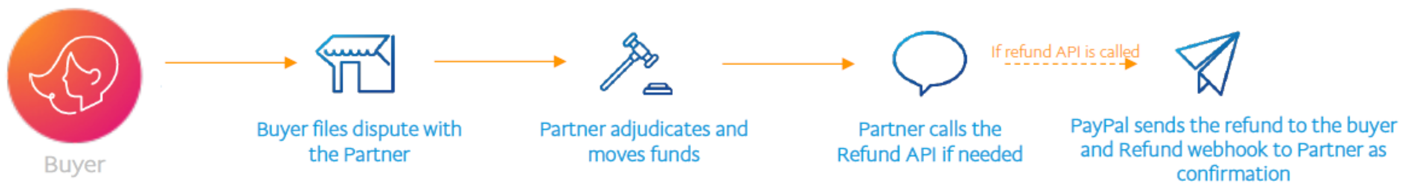
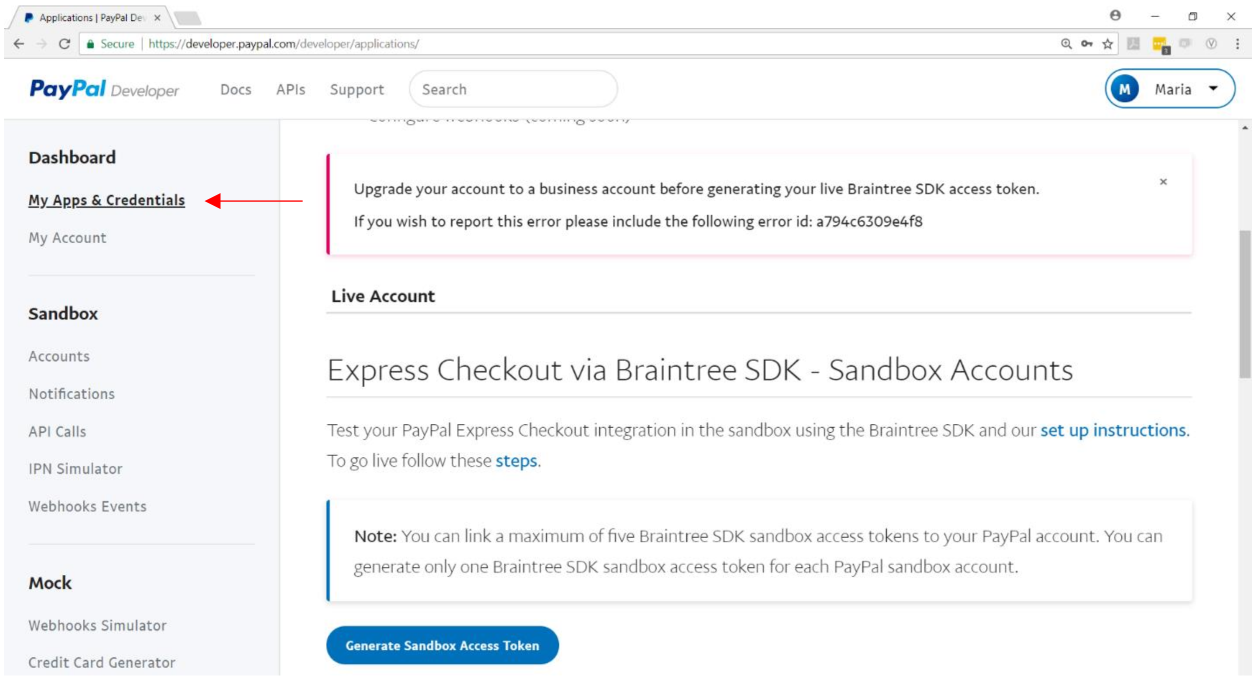

Buyer
Buyer files dispute with the Partner
Partner adjudicates and moves funds
Partner calls the Refund API if needed
If refund API is called
PayPal sends the refund to the buyer and Refund webhook to Partner as confirmation

Figure 2. Dispute API flow directly with merchant

3. Account configuration

Start by configuring the Sandbox account that you will be using during the testing of this API. To do that, go to www.developer.paypal.com and click on the option of *My Apps & Credentials*, beneath the *Dashboard* section in the left menu, see figure 3.



Upgrade your account to a business account before generating your live Braintree SDK access token.

If you wish to report this error please include the following error id: a794c6309e4f8

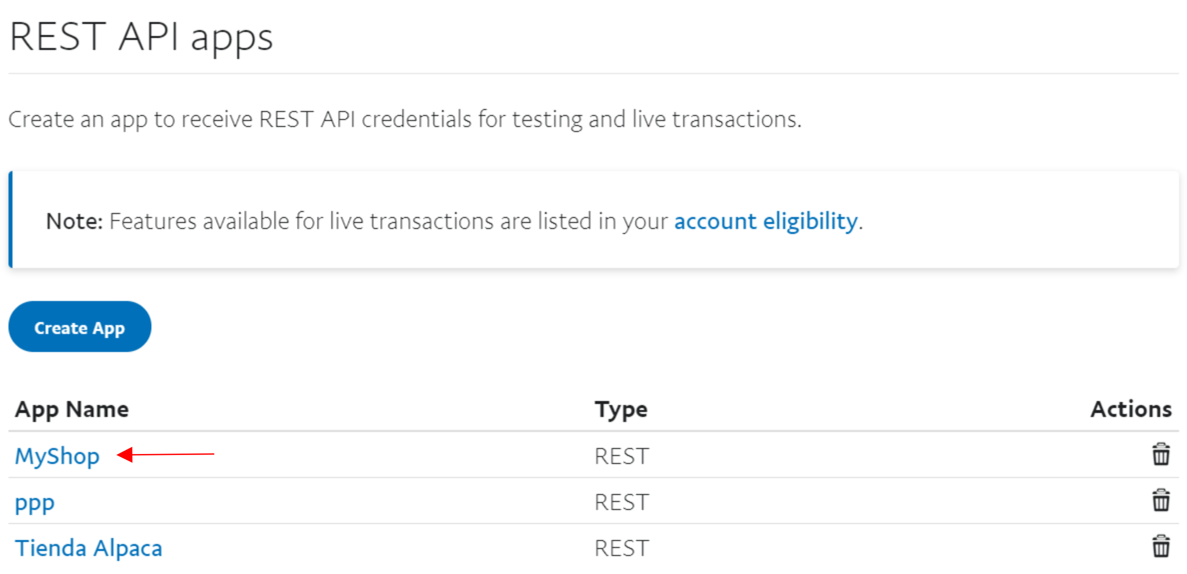
Express Checkout via Braintree SDK - Sandbox Accounts

Test your PayPal Express Checkout integration in the sandbox using the Braintree SDK and our set up instructions. To go live follow these steps.

Note: You can link a maximum of five Braintree SDK sandbox access tokens to your PayPal account. You can generate only one Braintree SDK sandbox access token for each PayPal sandbox account.

Figure 3. My Apps & Credentials

Then, go to *REST API apps* and select the one you will use to do the test as you can see in figure 4. If you do not have any app, please create one by clicking in the *Create App* button.


REST API apps

Create an app to receive REST API credentials for testing and live transactions.

Note: Features available for live transactions are listed in your account eligibility.

App Name	Type	Actions
MyShop	REST	🗑
ppp	REST	🗑
Tienda Alpaca	REST	🗑

Figure 4. Select your REST API app

Finally, in the *Sandbox App Setting* menu select the option of **Customer Disputes** and click on *Save*. With this, you will be able to do any test on the Sandbox environment.

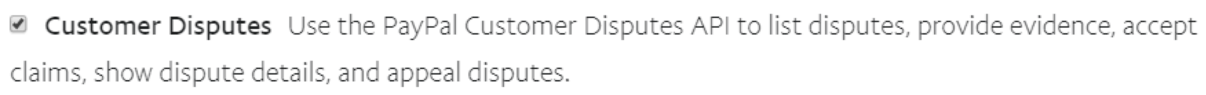
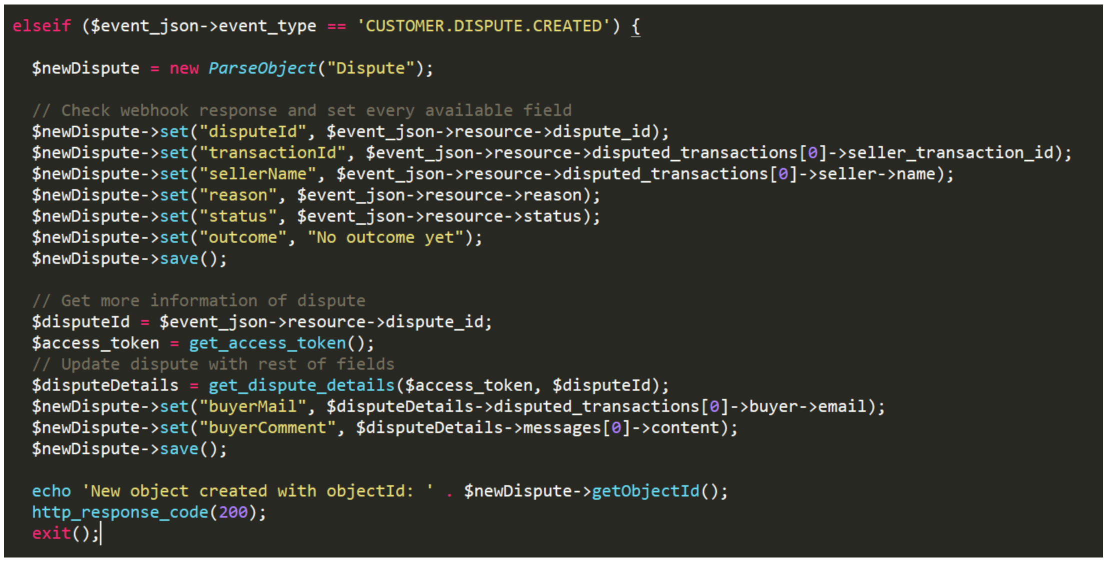

✓ **Customer Disputes** Use the PayPal Customer Disputes API to list disputes, provide evidence, accept claims, show dispute details, and appeal disputes.

Figure 5. Customer Disputes activation

4. Create a 'Webhook listener'

You can use 'Webhooks' to trigger the creation of a new object in your disputes database. First thing you should do is the code/setup of your webhook, by listening to the `CUSTOMER.DISPUTE.CREATED` event and triggering an action. In the example of figure 6, the webhook triggers the creation of a new object in the database, as well as the filling of dispute details according to the webhook response. After creating this file (in this example a php file was created), you must upload it to a secure server. This way, every time a dispute is created your server will be able to do both: listen the event and modify your database.

NOTE: The `CUSTOMER.DISPUTE.UPDATED` and `CUSTOMER.DISPUTE.RESOLVED` can trigger other updates or actions in the database.



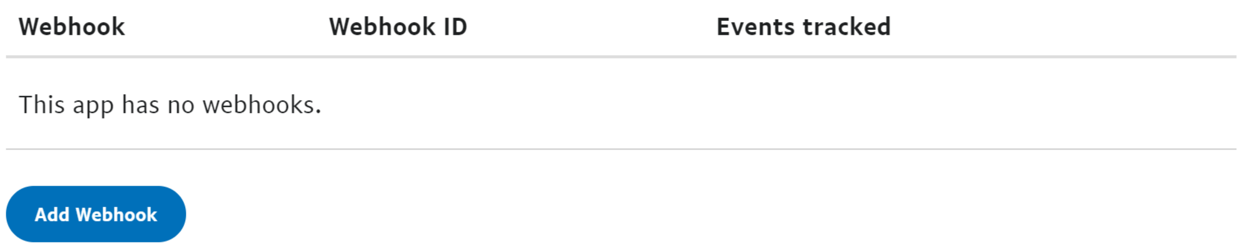
```php
elseif ($event_json->event_type == 'CUSTOMER.DISPUTE.CREATED') {

  $newDispute = new ParseObject("Dispute");

  // Check webhook response and set every available field
  $newDispute->set("disputeId", $event_json->resource->dispute_id);
  $newDispute->set("transactionId", $event_json->resource->disputed_transactions[0]->seller_transaction_id);
  $newDispute->set("sellerName", $event_json->resource->disputed_transactions[0]->seller->name);
  $newDispute->set("reason", $event_json->resource->reason);
  $newDispute->set("status", $event_json->resource->status);
  $newDispute->set("outcome", "No outcome yet");
  $newDispute->save();

  // Get more information of dispute
  $disputeId = $event_json->resource->dispute_id;
  $access_token = get_access_token();
  // Update dispute with rest of fields
  $disputeDetails = get_dispute_details($access_token, $disputeId);
  $newDispute->set("buyerMail", $disputeDetails->disputed_transactions[0]->buyer->email);
  $newDispute->set("buyerComment", $disputeDetails->messages[0]->content);
  $newDispute->save();

  echo 'New object created with objectId: ' . $newDispute->getObjectId();
  http_response_code(200);
  exit();
```

Figure 6. Webhook code – Created dispute

`CUSTOMER.DISPUTE.CREATED` *Webhook Response Example*

```
{
	"id": "WH-82963375VM412172U-5NX91127UN1954721",
	"create_time": "2016-06-09T20:39:36Z",
	"resource_type": "dispute",
	"event_type": "CUSTOMER.DISPUTE.CREATED",
	"summary": "A new dispute opened with Case # PP-000-001-341-393",
	"resource": {
		"disputed_transactions": [
			{
				"buyer_transaction_id": "9FN476350Y918024J",
				"seller_transaction_id": "45V487086B989905H",
				"seller_protection_eligible": true
			}
		],
		"reason": "UNAUTHORIZED",
		"create_time": "2016-06-09T20:38:52.000Z",
		"dispute_amount": {
			"currency_code": "USD",
			"value": "11.56"
		},
		"dispute_id": "PP-000-001-341-393",
		"status": "OPEN"
	},
	"links": [
		{
			"href": "https://10.24.121.37:14084/v1/notifications/webhooks-events
/WH-82963375VM412172U-5NX91127UN1954721",
			"rel": "self",
			"method": "GET",
			"encType": "application/json"
		},
		{
			"href": "https://10.24.121.37:14084/v1/notifications/webhooks-events
/WH-82963375VM412172U-5NX91127UN1954721/resend",
			"rel": "resend",
			"method": "POST",
			"encType": "application/json"
		}
	],
	"event_version": "1.0"
}
```

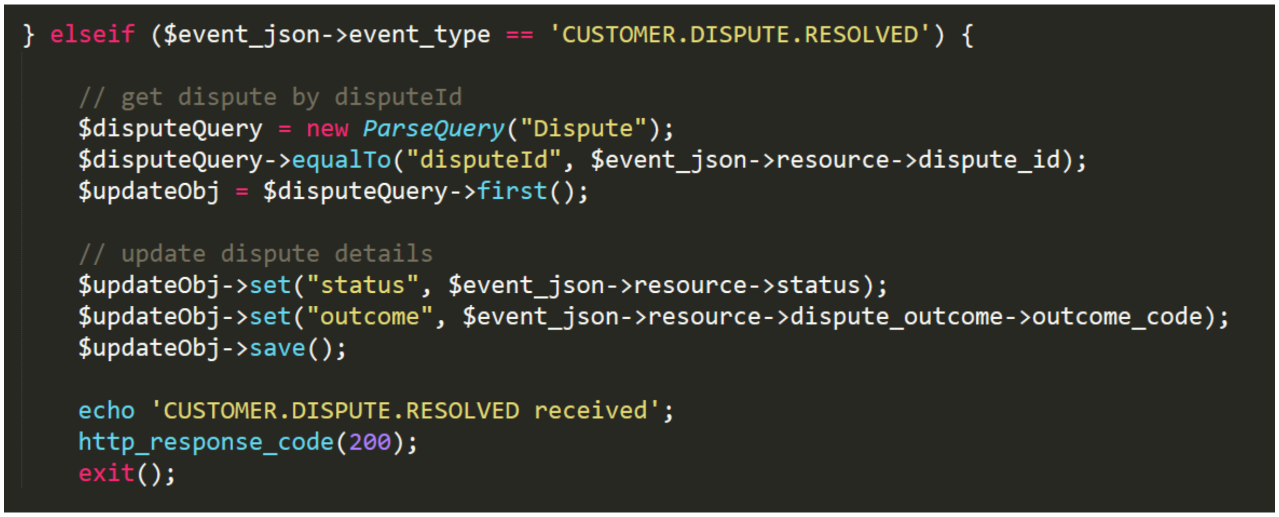
Set up your webhook listener in Sandbox to perform some testing. Go to developer.paypal.com and click on the option of *My Apps & Credentials*, beneath the *Dashboard* section in the left menu, see figure 3. Then, scroll to the *REST API apps* section and select the one you will use to do the test, as you can see in figure 4. Scroll until you find the *Add Webhook* button, see figure 7. Click on it, paste your webhook listener URL, select the events you want to listen to and save.



Webhook	Webhook ID	Events tracked
This app has no webhooks.		

Figure 7. Webhook setup in sandbox

Finally, once an event is triggered you can make a call to any other Dispute API or listen to other dispute related events, such as `CUSTOMER.DISPUTE.UPDATED` or `CUSTOMER.DISPUTE.RESOLVED`, and use those events to update your database, as shown in figure 8.



```
} elseif ($event_json->event_type == 'CUSTOMER.DISPUTE.RESOLVED') {

    // get dispute by disputeId
    $disputeQuery = new ParseQuery("Dispute");
    $disputeQuery->equalTo("disputeId", $event_json->resource->dispute_id);
    $updateObj = $disputeQuery->first();

    // update dispute details
    $updateObj->set("status", $event_json->resource->status);
    $updateObj->set("outcome", $event_json->resource->dispute_outcome->outcome_code);
    $updateObj->save();

    echo 'CUSTOMER.DISPUTE.RESOLVED received';
    http_response_code(200);
    exit();
```

Figure 8. Webhook code – Resolved dispute

`CUSTOMER.DISPUTE.UPDATED` *Webhook Response Example*

```
{
	"id": "WH-774363375VM412172U-5NX91127UN1954721",
	"create_time": "2016-12-09T10:39:36Z",
	"resource_type": "dispute",
	"event_type": "CUSTOMER.DISPUTE.UPDATED",
	"summary": "A dispute with Case # PP-000-001-001-001 got updated",
	"resource": {
		"disputed_transactions": [
			{
				"buyer_transaction_id": "9FN476350Y918024J",
				"seller_transaction_id": "45V487086B989905H",
				"seller_protection_eligible": true
			}
		],
		"reason": "UNAUTHORIZED",
		"create_time": "2016-12-09T10:39:36Z",
		"dispute_amount": {
			"currency_code": "USD",
			"value": "11.56"
		},
		"dispute_id": "PP-000-001-001-001",
		"status": "OPEN"
	},
	"links": [
		{
			"href": "https://10.24.121.37:14084/v1/notifications/webhooks-events
/WH-82963375VM412172U-5NX91127UN1954721",
			"rel": "self",
			"method": "GET",
			"encType": "application/json"
		},
		{
			"href": "https://10.24.121.37:14084/v1/notifications/webhooks-events
/WH-82963375VM412172U-5NX91127UN1954721/resend",
			"rel": "resend",
			"method": "POST",
			"encType": "application/json"
		}
	],
	"event_version": "1.0"
}
```

`CUSTOMER.DISPUTE.RESOLVED` *Webhook Response Example*

```
{
	"id": "WH-9M585274PE5867232-40C32040700472204",
	"create_time": "2016-06-07T22:56:55Z",
	"resource_type": "dispute",
	"event_type": "CUSTOMER.DISPUTE.RESOLVED",
	"summary": "A dispute was resolved with case # PP-000-001-001-001",
	"resource": {
		"disputed_transactions": [
			{
				"buyer_transaction_id": "22V23554K3057852G",
				"seller_transaction_id": "6C589962JJ4368547",
				"seller_protection_eligible": true
			}
		],
		"reason": "UNAUTHORIZED",
		"update_time": "2016-06-07T22:56:19.000Z",
		"create_time": "2016-06-07T22:53:25.000Z",
		"dispute_amount": {
			"currency_code": "USD",
			"value": "9.98"
		},
		"dispute_id": "PP-000-001-001-001",
                    "dispute_outcome": {
                            "outcome_code": "BUYER_FAVOUR"
                    },
                    "status": "RESOLVED"
        },
        "links": [
                    {
                            "href": "https://10.24.121.37:14084/v1/notifications/webhooks-events
/WH-9M585274PE5867232-40C32040700472204",
                            "rel": "self",
                            "method": "GET",
                            "encType": "application/json"
                    },
                    {
                            "href": "https://10.24.121.37:14084/v1/notifications/webhooks-events
/WH-9M585274PE5867232-40C32040700472204/resend",
                            "rel": "resend",
                            "method": "POST",
                            "encType": "application/json"
                    }
        ],
        "event_version": "1.0"
}
```

For further documentation on webhooks, please ask for the Webhooks guide.

5. Obtain details of the transaction

There are ten different API calls for Customer Disputes, plus the one that includes the creation of the Access Token.

I. Create token
II. List disputes
III. Show dispute details
IV. Accept claim
V. Settle dispute
VI. Appeal dispute
VII. Escalate dispute to a claim
VIII. Make offer to resolve dispute
IX. Provide evidence
X. Update dispute state
XI. Send message to other party

The very first thing you must do is create a Token to have the authentication for the rest of the calls.

- **Create Token**

Authentication

PayPal APIs are RESTful APIs and utilize Oauth for authentication. For help in making your first API call against PayPal APIs, please go to https://developer.paypal.com/docs/integration/direct/make-your-first-call/.

In the details for each API, you will find the appropriate endpoints for the PayPal Sandbox as well as for Production.

The GetAccessToken API is utilized to retrieve an access token for future API calls. Detailed API specification can be found at:

https://developer.paypal.com/docs/api/#authentication--headers

API Endpoints (POST)

- **Testing Server**: https://api.sandbox.paypal.com/v1/oauth2/token
- **Production Server**: https://api.paypal.com/v1/oauth2/token

Important API Request Headers

The request headers below, must be set according to the following definitions:

- **Authorization**: Set to 'Basic (Base64 Encode {<Client ID>: <Secret>})' where Client ID / Secret is the Client ID / Secret of the REST App associated with the API caller account.

Important Request Parameters

The request parameters below, must be set according to the following definitions:

- **grant_type**: Must be set to 'client_credentials' to acquire an access token for this client.

Important Response Parameters

The response parameters below are important for this integration:

- **access_token**: This is the access token which will allow your API caller to make future API calls. It has an expiration time as defined by expires_in, and can be utilized until that expiration without renewal.
- **expires_in**: This defines when the access_token will expire, in seconds. By default, this is set to 8 hours, but PayPal may change this at some point in the future.

<u>Example API Request</u>

```
curl -k -v https://api.sandbox.paypal.com/v1/oauth2/token -H 'Accept: application/json' -H
'Accept-Language: en_US' -u AfxcT3emGyraHH3dVSXAicmJoK8fb9E8PfVSX2RiyfxU-
iQ6YXewrBrRS_mmOu8hkDfKrLAS03Jd1qGe:EEKa8N398icFGL6LIE4Rvz8hr1KKK1W_XFf0__adJB3zT-
0i0h04RdMkBJ7T2KSeJc4ljiWMZxrNQS4X -d grant_type=client_credentials
```

<u>Example API Response</u>

```
{
   "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":"2016-07-29T06:12:06Zm9Ziq7cHq9hYWFENcvAsOVlHbtNnIV90bikkUIsZElg",
   "access_token":"A101.g9Tx8ExiTN9X4qI0Fu0rFO1FSaEKWVqi4ogT68AAa-
XK9OtYhPqN6jIgTYsOxudt.s4m6n8wK1syCugzg1sbjS5rLVSi",
   "token_type":"Bearer",
   "app_id":"APP-80W284485P519543T",
   "expires_in":31462
}
```

Once you have obtained your Bearer token, you can have access to all the information about any transaction you want and/or dispute.

- **List Disputes**

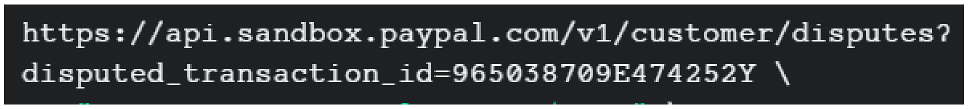
The first API you will use is the List Dispute, which shows the information related to a specific transaction id.

<u>API Endpoints (GET)</u>

- **Testing Server**:

https://api.sandbox.paypal.com/v1/customer/disputes?dispute_transaction_id:***transactionId***

Example:



```
https://api.sandbox.paypal.com/v1/customer/disputes?
disputed_transaction_id=965038709E474252Y \
```

- **Production Server**:

https://api.paypal.com/v1/ customer/disputes?dispute_transaction_id:***transactionId***

Important API Request Headers

The request headers below, must be set according to the following definitions:

- **Authorization:** Set the option on ‘Bearer <Access-Token>’. This variable Access-Token is the token returned from the /v1/oauth2/token API.
- **Accept**: Set to ‘application/json’.

API Response

The result is a JSON response body that lists disputes with the `dispute_id` (this is the most important variable from this API), `reason`, `status`, `dispute_amount`, `create_time`, and `update_time` fields for each dispute.

Example of response:

```
{
    "items": [
        {
            "dispute_id": "PP-000-042-638-566",
            "create_time": "2018-04-02T21:12:41.000Z",
            "update_time": "2018-04-22T21:23:03.000Z",
            "status": "RESOLVED",
            "dispute_amount": {},
            "links": [
                {
                    "href":  "https://api.sandbox.paypal.com/v1/customer/disputes/PP-000-042-638-
566",
                    "rel": "self",
                    "method": "GET"
                }
            ]
        }
    ],
    "links": [
        {
            "href":
"https://api.sandbox.paypal.com/v1/customer/disputes?disputed_transaction_id=5FH949350M9729710",
            "rel": "self",
            "method": "GET",
            "encType": "application/json"
        },
        {
            "href":
"https://api.sandbox.paypal.com/v1/customer/disputes?disputed_transaction_id=5FH949350M9729710",
            "rel": "first",
            "method": "GET",
            "encType": "application/json"
        }
    ]
}
```

- ## Show dispute details

With this API you can get all the details of a specific dispute using the `dispute_id`, that you already obtained from the last API call.

API Endpoints (GET)

- **Testing Server**:

https://api.sandbox.paypal.com/v1/customer/disputes/***dispute_id***

Example:

```
https://api.sandbox.paypal.com/v1/customer/disputes/PP-
000-003-648-191 \
```

- **Production Server**:

https://api.paypal.com/v1/customer/disputes/***dispute_id***

Important API Request Headers

The request headers below, must be set according to the following definitions:

- **Authorization:** Set the option on 'Bearer <Access-Token>'. This variable Access-Token is the token returned from the /v1/oauth2/token API.
- **Accept**: Set to 'application/json'.

API Response

The result JSON response body will show all the dispute details.

Example of response:

```
{
    "dispute_id": "PP-000-042-638-566",
    "create_time": "2018-04-02T21:12:41.000Z",
    "update_time": "2018-04-22T21:23:03.000Z",
    "disputed_transactions": [
        {
            "buyer_transaction_id": "5FH949350M9729710",
            "seller_transaction_id": "0UW81631T1635615Y",
            "create_time": "2018-04-02T21:10:39.000Z",
            "transaction_status": "COMPLETED",
            "gross_amount": {
                "currency_code": "MXN",
                "value": "198.00"
            },
            "buyer": {
                "email": "nadadora.comprador@prueba.com",
                "name": "Maria Nieto"
            },
            "seller": {
                "email": "vendedor.nadadora@prueba.com",
                "merchant_id": "JJM24XM24B7CA",
                "name": "Tienda  Uno's Test Store"
            },
            "items": [
                {
                    "item_id": "001"
                }
            ],
            "seller_protection_eligible": true
        }
    ],
    "reason": "MERCHANDISE_OR_SERVICE_NOT_RECEIVED",
    "status": "RESOLVED",
    "dispute_amount": {
        "currency_code": "MXN",
        "value": "198.00"
    },
    "dispute_life_cycle_stage": "INQUIRY",
    "dispute_channel": "INTERNAL",
    "messages": [
        {
            "posted_by": "BUYER",
            "time_posted": "2018-04-02T21:12:43.000Z",
            "content": "Prueba de Dispute para Settle API"
        }
    ],
    "links": [
        {
            "href": "https://api.sandbox.paypal.com/v1/customer/disputes/PP-000-042-638-566",
            "rel": "self",
            "method": "GET"
        }
    ]
}
```

6. Response to the dispute

With these API calls you can get and change the status of a dispute, accept, settle in either buyer's or seller's favor or send files as evidence. You will have to use the `dispute_id` that you obtained from the List Disputes or Show Dispute Details API.

• Accept claim

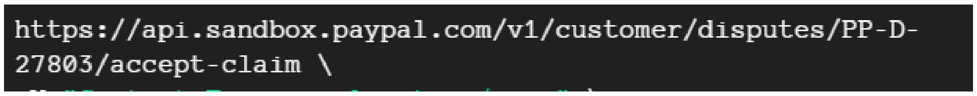
This API allows the merchant to accept liability for a claim, which closes the dispute in the customer's favor. PayPal refunds money to the customer from the merchant's account.

API Endpoints (POST)

- **Testing Server**:

https://api.sandbox.paypal.com/v1/customer/disputes/***dispute_id/accept-claim***

Example:


```
https://api.sandbox.paypal.com/v1/customer/disputes/PP-D-
27803/accept-claim \
```

- **Production Server**:

https://api.paypal.com/v1/customer/disputes/***dispute_id/accept-claim***

Important API Request Headers

The request headers below, must be set according to the following definitions:

- **Authorization:** Set the option on ‘Bearer <Access-Token>’. This variable Access-Token is the token returned from the /v1/oauth2/token API.
- **Accept**: Set to ‘application/json’.

Important request parameters

I. **dispute_id:** The ID of the dispute for which to accept a claim.

API Response

The result JSON response body includes a link to the dispute and a 200 status.

Example

Example of response:

```
{
    "links": [
        {
            "href": "https://api.sandbox.paypal.com/v1/customer/disputes/PP-000-042-645-315",
            "rel": "detail",
            "method": "GET",
            "encType": "application/json"
        }
    ]
}
```

- ## Settle dispute

Sandbox only. This API settles a dispute in either the buyer's or the seller's favor. Merchants can make this call in the sandbox to complete end-to-end dispute resolution testing. To make this call, the dispute `status` must be `UNDER_REVIEW`.

API Endpoints (POST)

- **Testing Server**:

https://api.sandbox.paypal.com/v1/customer/disputes/***dispute_id/adjudicate***

Example:

```
https://api.sandbox.paypal.com/v1/customer/disputes/PP-D-
27803/adjudicate \
```

Important API Request Headers

The request headers below, must be set according to the following definitions:

- **Authorization:** Set the option on 'Bearer <Access-Token>'. This variable Access-Token is the token returned from the /v1/oauth2/token API.
- **Accept**: Set to 'application/json'.

Important Request Parameters

- **dispute_id:** The ID of the dispute to settle.

Important Request Body Parameters

- **adjudication_outcome:** The outcome of the adjudication. Allowed values: `BUYER_FAVOR, SELLER_FAVOR`.

API Response

The result JSON response body includes a link to the dispute.

- ### Appeal dispute

To appeal a dispute, use the appeal link in the HATEOAS links from the show dispute details response. If this link does not appear, you cannot appeal the dispute. You can send evidence with this API.

API Endpoints (POST)

- **Testing Server**:

https://api.sandbox.paypal.com/v1/customer/disputes***/dispute_id*** /***appeal***

Example:

```
https://api.sandbox.paypal.com/v1/customer/disputes/PP-D-
27803/appeal \
```

- **Production Server**:

https://api.paypal.com/v1/customer/disputes/***dispute_id/appeal***

Important API Request Headers

The request headers below, must be set according to the following definitions:

- **Authorization:** Set the option on ‘Bearer <Access-Token>’. This variable Access-Token is the token returned from the /v1/oauth2/token API.
- **Accept**: Set to ‘multipart/related’.

Important request parameters

- **evidence:** An array of evidences for the dispute.

API Response

The result is a JSON response body that includes a link to the dispute.

Important response parameters

- **links**: An array of request-related HATEOAS links.

- ## Escalate dispute to a claim

With this API you can escalates the dispute to a PayPal claim. To make this call, the stage in the dispute lifecycle must be `INQUIRY`.

API Endpoints (POST)

- **Testing Server**:

https://api.sandbox.paypal.com/v1/customer/*disputes*/***dispute_id/escalate***

Example:

```
https://api.sandbox.paypal.com/v1/customer/disputes/PP-000-000-
651-454/escalate \
```

- **Production Server**:

https://api.paypal.com/v1/customer/disputes/***dispute_id/escalate***

Important API Request Headers

The request headers below, must be set according to the following definitions:

- **Authorization:** Set the option on ‘Bearer <Access-Token>’. This variable Access-Token is the token returned from the /v1/oauth2/token API.
- **Accept**: Set to ‘multipart/related’

Important request parameters

- **note:** The notes about the escalation of the dispute to a claim.

API Response

The result is a JSON response body that includes a link to the dispute.

Important response parameters

- **links**: An array of request-related HATEOAS links

- ### Make offer to resolve dispute

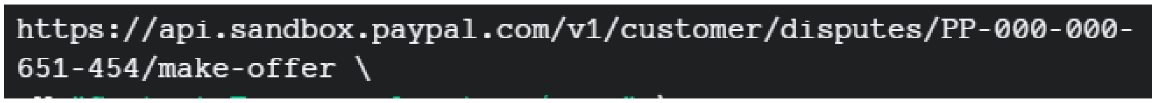
With this API the merchant can make an offer to the other party to resolve the dispute. To make this call, the stage in the dispute lifecycle must be `INQUIRY`. If the customer accepts the offer, PayPal automatically makes the refund.

API Endpoints (POST)

- **Testing Server**:

https://api.sandbox.paypal.com/v1/customer/*disputes*/***dispute_id/make-offer***

Example:



```
https://api.sandbox.paypal.com/v1/customer/disputes/PP-000-000-
651-454/make-offer \
```

- **Production Server**:

https://api.paypal.com/v1/customer/disputes/***dispute_id/make-offer***

Important API Request Headers

The request headers below, must be set according to the following definitions:

- **Authorization:** Set the option on ‘Bearer <Access-Token>’. This variable Access-Token is the token returned from the /v1/oauth2/token API.
- **Accept**: Set to ‘application/json’.

Important request parameters

- **note:** The notes about the escalation of the dispute to a claim.
- **offer_amount:** The amount proposed to resolve the dispute.
- **return_shippping_address:** The return address for the item. For the `MERCHANDISE_OR_SERVICE_NOT_AS_DESCRIBE` dispute reason.
- **invoice_id:** The merchant-provided ID of the invoice for the refund.
- **offer_type:** The type of offer that the merchant proposes for the dispute.

API Response

The result is a JSON response body that includes a link to the dispute.

Important response parameters

- **links**: An array of request-related HATEOAS links

- ## Provide evidence

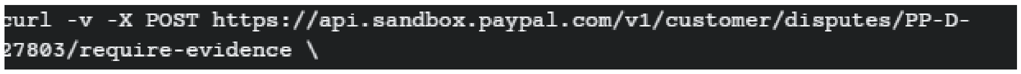
A merchant can provide evidence for disputes with the `WAITING_FOR_SELLER_RESPONSE` status while customers can provide evidence for disputes with the `WAITING_FOR_BUYER_RESPONSE` status. Evidence can be a proof of delivery or proof of refund document or receipts, which can include logs. A proof of delivery document includes a tracking number, while a proof of refund document includes a refund ID.

API Endpoints (POST)

- **Testing Server**:

https://api.sandbox.paypal.com/v1/customer/disputes/***dispute_id*/*provide-evidence***

Example:



```
curl -v -X POST https://api.sandbox.paypal.com/v1/customer/disputes/PP-D-
27803/require-evidence \
```

- **Production Server**:

https://api.paypal.com/v1/customer/disputes/***dispute_id*****/*****provide-evidence***

Important API Request Headers

The request headers below must be set according to the following definitions.

- **Authorization:** Set the option on 'Bearer <Access-Token>'. This variable Access-Token is the token returned from the /v1/oauth2/token API.
- **Accept**: Set to 'multipart/related'

Important Request Data Parameters

- **evidence:** A file with evidence.

API Response

The result is a JSON response body that includes a link to the dispute.

Important response parameters

- **links**: An array of request-related HATEOAS links.

Update dispute state

Sandbox only. With this API call you can update the state of a dispute, using the `dispute_id.` To make this call, the `status` must be `UNDER_REVIEW`. You can change the state to:

- `WAITING_FOR_BUYER_RESPONSE`
- `WAITING_FOR_SELLER_RESPONSE`

This state change enables either the buyer or seller to submit evidence for the dispute. Depending on the action, the states updates as you can see in the next table.

Action	State
SELLER_EVIDENCE	WAITING_FOR_SELLER_RESPONSE
BUYER_EVIDENCE	WAITING_FOR_BUYER_RESPONSE

Table 1. Actions and their states.

API Endpoints (POST)

- **Testing Server**:

https://api.sandbox.paypal.com/v1/customer/disputes/***dispute_id*****/*****require-evidence***

Example:

```
https://api.sandbox.paypal.com/v1/customer/disputes/PP-D-
27803/require-evidence \
```

Important API Request Headers

The request headers below, must be set according to the following definitions:

- **Authorization:** Set the option on ‘Bearer <Access-Token>’. This variable Access-Token is the token returned from the /v1/oauth2/token API.
- **Accept**: Set to ‘application/json’.

Important Request Parameters

- **dispute_id:** The ID of the dispute that requires evidence.

Important Request Body Parameters

- **action:** The action to be completed. Allowed values: `BUYER_EVIDENCE`, `SELLER_EVIDENCE`.

API Response

The result is a JSON response body that includes a link to the dispute.

Important response parameters

- **links**: An array of request-related HATEOAS links.

- ## Send message to other party

This API sends a message to the other party in the dispute.

API Endpoints (POST)

- **Testing Server**:

https://api.sandbox.paypal.com/v1/customer/disputes/***dispute_id*****/send-message**

Example:

```
https://api.sandbox.paypal.com/v1/customer/disputes/PP-000-000-
651-454/send-message \
```

- **Production Server**:

https://api.paypal.com/v1/customer/disputes/***dispute_id/send-message***

Important API Request Headers

The request headers below, must be set according to the following definitions:

- **Authorization:** Set the option on 'Bearer <Access-Token>'. This variable Access-Token is the token returned from the /v1/oauth2/token API.
- **Accept**: Set to 'application/json'.

Important request parameters

- **dispute_id:** The ID of the dispute for which to accept a claim
- **message:** The message that is send by the merchant to the other party.

API Response

The result is a JSON response body that includes a link to the dispute.

Important response parameters

- **links**: An array of request-related HATEOAS links.

Example API Request

```
curl -X POST \
  https://api.sandbox.paypal.com/v1/customer/disputes/PP-000-042-647-765/send-message \
  -H 'Cache-Control: no-cache' \
  -H 'Content-Type: application/json' \
  -H 'Postman-Token: 79eb42e9-c659-8995-ad6d-d34c64f70ee0' \
  -d '{
  "message": "Shipment is in progress."
}
```

7. Example, Official Docs and Postman Collection

For more information, visit these links:

- ✓ https://developer.paypal.com/docs/api/customer-disputes/
- ✓ https://developer.paypal.com/docs/integration/direct/customer-disputes/
- ✓ https://developer.paypal.com/docs/marketplaces/disputes/integration-guide/

Some coded examples:

- ✓ An example of a case: http://www.tiendaejemplomx.com/gpozos/webhook/ss_disputes/mis-disputas.php
- ✓ The code of the example case: https://github.com/gpozos-pp/disputes
- ✓ The 'Postman collection' with all the API calls that are explained in this document: https://www.getpostman.com/collections/1909ee4b975bd929e47b