Accept Payments via Payment Links | Developer Documentation

Accept Payments via Payment Links

Updated: Nov 14, 2025
This feature is not publicly available yet. Please reach out to whatsappindia-bizpayments-support@meta.com to know more.
Your businesses can enable customers to pay for their orders by bringing in all the payment methods supported on your platform to WhatsApp. Businesses can send customers invoice(`order_details`) messages, then get notified about payment status updates via webhook notifications from Payment Gateway.

Overview

Currently, customers browse business catalogs, add products to cart, and send orders in with our set of commerce messaging solutions, which includes Single Product Message, Multi Product Message, and Product Detail Page.
With the WhatsApp Messaging API, businesses can send customers a bill to complete the order with one of the supported payment instrument.

How It Works

The business must send an `order_details` message for the consumer to initiate payment. This type of message is a new type of interactive message, which always contains the same 4 main components: **header**, **body**, **footer**, and **action**. Inside the **action** component, the business includes all the information needed for the customer to complete their payment.

Each `order_details` message contains a unique `reference_id` provided by the business, and that unique number is used throughout the flow to track the order. This `reference_id` is used to generate the payment link from Payment Gateway.
Once the message is sent, the business waits for a payment or transaction status updates directly from Payment Gateway. Upon receiving payment signal for an order, business should relay this payment signal to consumer client through interactive order status(`order_status`) message. Updating user about the payment signal is important as this message updates the order details message and order details view for the consumer reflecting the order confirmation from merchant. This is shown with an example in subsequent sections.

Purchase Flow in App

In the WhatsApp customer app, the purchase flow has the following steps:
Customer sends an order with selected products to the business or business identifies the products that the customer has shown interest to purchase.
After receiving the order or identifying the product, if a business accepts payment methods other than UPI, such as credit cards and payment wallets, etc. then business will send a message to the user to get their preferred payment method for the order.
Image
When consumers want to pay using other payment method option, the business should generate the payment link by calling Payment Gateway by providing the unique “reference-id” and other information like amount, validity etc, then business can use the generated payment link to construct the order details message and send to the consumer.
ImageImage
When the consumer taps the Pay now/continue button, consumer will be redirected to the payment link within specially designed In-App browser to present with the list of supported payment options such as credit card, debit card, wallet or UPI apps. Consumers can choose any one of the payment option to pay for the order.
The following is a sample payment link redirect within In-App Browser accepting various payment methods like credit, debit, wallet and UPI apps.
ImageImage
Once the payment is complete, the business will receive a notification from Payment Gateway and the business needs to send order status updates to the consumer client notifying consumers about the progress on their order, this will update the order details message CTAs, Order details screen and Order status. The order status should contain the matching “reference-id” of order details.
ImageImage

Integration Steps

The steps outlined below assume that the business is about to send order details message to consumer client.
The following sequence diagram demonstrates the typical integration flow for WA Payments API:
Image

Step 1: Get Payment Link from Payment Gateway

Once the consumer has expressed their interest to purchase an item using payment link. Business needs to call payment gateway with necessary information like reference-id, amount and validity to generate the payment link. Following is a sample payment link:

```
https://rzp.io/i/rNiAagU8y
```

Business needs to use the same reference-id, amount and expiration in invoice(`order_details`) interactive message.

Step 2: Assemble the Interactive Object

To send an `order_details` message, businesses must assemble an interactive object of type `order_details` with the following components:

Object	Description
`type` object	**Required.** Must be “order_details”.
`header` object	**Optional.** Header content displayed on top of a message. If a header is not provided, the API uses an image of the first available product as the header
`body` object	**Required.** An object with the body of the message. The object contains the following field: `text` string **Required** if `body` is present. The content of the message. Emojis and markdown are supported. Maximum length is 1024 characters
`footer` object	**Optional.** An object with the footer of the message. The object contains the following fields: `text` string **Required** if `footer` is present. The footer content. Emojis, markdown, and links are supported. Maximum length is 60 characters
`action` object	**Required.** An action object you want the user to perform after reading the message. This action object contains the following fields: `name` string **Required**. Must be “review_and_pay” `parameters` object See Parameters Object for information

Parameters Object

Object	Description
`reference_id` string	**Required.** Unique identifier for the order or invoice provided by the business. It is case sensitive and cannot be an empty string and can only contain English letters, numbers, underscores, dashes, or dots, and should not exceed 35 characters. The `reference_id` must be unique for each `order_details` message for the same business. If the partner would like to send multiple order_details messages for the same order, invoice, etc. it is recommended to include a sequence number in the `reference_id` (for example, <order-or-invoice-id>-<sequence-number>) to ensure `reference_id` uniqueness.
`type` object	**Required.** The type of goods being paid for in this order. Current supported options are `digital-goods` and `physical-goods`
`beneficiaries` array	**Required for shipped physical-goods.** An array of beneficiaries for this order. A beneficiary is an intended recipient for shipping the physical goods in the order. It contains the following fields: Beneficiary information isn't shown to users but is needed for legal and compliance reasons. `name` string **Required.** Name of the individual or business receiving the physical goods. Cannot exceed 200 characters `address_line1` string **Required.** Shipping address (Door/Tower Number, Street Name etc.). Cannot exceed 100 characters `address_line2` string **Optional.** Shipping address (Landmark, Area, etc.). Cannot exceed 100 characters `city` string **Required.** Name of the city. `state` string **Required.** Name of the state. `country` string **Required.** Must be "India". `postal_code` string **Required.** 6-digit zipcode of shipping address.
`payment_type`	**Required.** Must be "upi".
`payment_settings`	**Required.** See Payment Settings Object for more details.
`currency`	**Required.** The currency for this order. Currently the only supported value is `INR`.

Object	Description
`total_amount` object	**Required.** The `total_amount` object contains the following fields: `offset` integer **Required.** Must be `100` for `INR`. `value` integer **Required.** Positive integer representing the amount value multiplied by offset. For example, ₹12.34 has value 1234. `total_amount.value` must be equal to `order.subtotal.value` + `order.tax.value` + `order.shipping.value` - `order.discount.value`.
`order` object	**Required.** See order object for more information.

Payment Setting Object

Object	Description
`type` string	**Required.** Must be `payment_link`.
`payment_link` object	**Required.** Refer Payment Link Object for more information.

Payment Link Object

Object	Description
`uri` string	**Required.** A valid payment link generated through payment gateway. Generated payment links domains needs to be enabled to accept payments. Please reach out to whatsappindia-bizpayments-support@meta.com to know more.
`success_url` string	**Optional.** The flow terminated with success status, when success_url is hit.
`cancel_url` string	**Optional.** The flow ends with failure, when the cancel_url is triggered.

Order Object

Object	Description
`status` string	**Required.** Only supported value in the `order_details` message is `pending`. In an `order_status` message, `status` can be: `pending`, `captured`, or `failed`.

Object	Description
`type` string	**Optional.** Only supported value is `quick_pay`. When this field is passed in we hide the “Review and Pay” button and only show the “Pay Now” button in the order details bubble.
`items` object	**Required.** An object with the list of items for this order, containing the following fields: `retailer_id` string **Optional.** Content ID for an item in the order from your catalog. `name` string **Required.** The item’s name to be displayed to the user. Cannot exceed 60 characters `image` object **Optional.** Custom image for the item to be displayed to the user. See item image object for information Using this image field will limit the items array to a maximum of 10 items and this cannot be used with `retailer_id` or `catalog_id`. `amount` amount object with value and offset -- refer total amount field above **Required.** The price per item `sale_amount` amount object **Optional.** The discounted price per item. This should be less than the original amount. If included, this field is used to calculate the subtotal amount `quantity` integer **Required.** The number of items in this order, this field cannot be decimal has to be integer. `country_of_origin` string **Required** if `catalog_id` is not present. The country of origin of the product `importer_name` string **Required** if `catalog_id` is not present. Name of the importer company `importer_adress` string **Required** if `catalog_id` is not present. Address of importer company
`subtotal` object	**Required.** The value **must be equal** to sum of `order.amount.value` * `order.amount.quantity`. Refer to `total_amount` description for explanation of `offset` and `value` fields The following fields are part of the `subtotal` object: `offset` integer **Required.** Must be `100` for `INR` `value` integer **Required.** Positive integer representing the amount value multiplied by offset. For example, ₹12.34 has value 1234

Object	Description
`tax` object	**Required.** The tax information for this order which contains the following fields: `offset` integer **Required.** Must be `100` for `INR` `value` integer **Required.** Positive integer representing the amount value multiplied by offset. For example, ₹12.34 has value 1234 `description` string **Optional.** Max character limit is 60 characters
`shipping` object	**Optional.** The shipping cost of the order. The object contains the following fields: `offset` integer **Required.** Must be `100` for `INR` `value` integer **Required.** Positive integer representing the amount value multiplied by offset. For example, ₹12.34 has value 1234 `description` string **Optional.** Max character limit is 60 characters
`discount` object	**Optional.** The discount for the order. The object contains the following fields: `offset` integer **Required.** Must be `100` for `INR` `value` integer **Required.** Positive integer representing the amount value multiplied by offset. For example, ₹12.34 has value 1234 `description` string **Optional.** Max character limit is 60 characters `discount_program_name` string **Optional.** Text used for defining incentivised orders. If order is incentivised, the merchant needs to define this information. Max character limit is 60 characters
`catalog_id` object	**Optional.** Unique identifier of the Facebook catalog being used by the business. If you do not provide this field, you must provide the following fields inside the items object: `country_of_origin`, `importer_name`, and `importer_address`
`expiration` object	**Optional.** Expiration for that order. Business must define the following fields inside this object: `timestamp` string – UTC timestamp in seconds of time when order should expire. Minimum threshold is 300 seconds `description` string – Text explanation for expiration. Max character limit is 120 characters

Item Image Object

Object	Description
`link` string	**Required.** A link to the image that will be shown to the user. Must be an `image/jpeg` or `image/png` and 8-bit, RGB or RGBA. Follows same requirements as image in media

The `parameters` value is a stringified JSON object.
By the end, the interactive object should look something like this for a catalog-based integration:
The `parameters` value is a stringified JSON object.
For a non-catalog based integration i.e. when catalog-id is not present, an example payload looks as follows:

Step 3: Add Common Message Parameters

Once the interactive object is complete, append the other parameters that make a message: `recipient_type`, `to`, and `type`. Remember to set the `type` to `interactive`.

These are parameters common to all message types.

Step 4:Make a POST Call to Messages Endpoint

Make a POST call to the `/[PHONE_NUMBER_ID]/messages` endpoint with the `JSON` object you have assembled. If your message is sent successfully, you get the following response:

Errors

WhatsApp Payments Terms of Service Acceptance Pending

If you see the following error, accept the WhatsApp Payments terms of service using the link provided in the error message before trying again.

For all other errors that can be returned and guidance on how to handle them, see WhatsApp Cloud API, Error Codes.

Step 5: Consumer Pays for the Order

Consumers can pay using WhatsApp payment method or using any UPI supported app that is installed on the device.

Step 6: Get Notified About Transaction Status Updates from payment gateway

Businesses receive updates to the invoice via payment gateway webhooks, when the status of the user-initiated transaction changes. The unique identifier reference-id passed in `order_details` message can be used to map the transaction to the consumer invoice or interactive order details message.

Step 7: Update order status

Upon receiving transaction signals from payment gateway through webhook, the business must update the order status to keep the user up to date. Currently we support the following order status values:
Image

Value	Description
`pending`	User has not successfully paid yet
`processing`	User payment authorized, merchant/partner is fulfilling the order, performing service, etc.
`partially-shipped`	A portion of the products in the order have been shipped by the merchant
`shipped`	All the products in the order have been shipped by the merchant
`completed`	The order is completed and no further action is expected from the user or the partner/merchant
`canceled`	The partner/merchant would like to cancel the `order_details` message for the order/invoice. The status update will fail if there is already a `successful` or `pending` payment for this `order_details` message

Typically businesses update the `order_status` using either the WhatsApp payment status change notifications or their own internal processes. To update `order_status`, the partner sends an `order_status` message to the user.
The following table describes the returned values:

Value	Description
`reference_id`	The ID provided by the partner in the `order_details` message
`status`	The new order `status`
`description`	Optional text for sharing status related information in `order_details`. Could be useful while sending cancellation. Max character limit is 120 characters

Merchant should always post this order-status message to consumer after receiving transaction updates for an order. As the order_details message and order details screen experience is tied to to the order status updates.

Step 8: Reconcile Payments

Businesses should use their bank statements to reconcile the payments using the `reference_id` provided in the `order_details` messages.

Checklist for Integrated Merchants

Ensure that `order_status` message is send to consumer informing them about updates to an order after receiving transaction updates for an order.
Ensure the merchant is verified and WABA contact is marked with a verified check.
Verify the WABA is mapped to appropriate merchant initiated messaging tier(1k, 10k and 100k per day)
Merchant should list the customer support information in the profile screen incase consumer wants to report any issues.

Revision #5
Created 2026-04-01 14:50:57 UTC by New Admin
Updated 2026-04-06 17:51:22 UTC by New Admin