bKash Payment Gateway : Checkout and Refund

Please complete the signup at Merchant Integration Portal to proceed with the next steps.

https://pgw-integration.bkash.com/sign-up

1.Developer Portal (detail Product, workflow, API information):
https://developer.bka.sh/docs/checkout-url-process-overview

2. Required APIs -

Grant Token : https://developer.bka.sh/docs/grant-token-1
Rrefresh Token : https://developer.bka.sh/docs/refresh-token-1

Create Payment : https://developer.bka.sh/docs/create-payment-2

Execute Payment : https://developer.bka.sh/docs/execute-payment-2

Query Payment : https://developer.bka.sh/docs/query-payment-1

Search :https://developer.bka.sh/docs/search-transaction-1

Refund : https://developer.bka.sh/docs/refund-transaction-1

Please note, API timeout of 30sec should be set for all the APIs.

3. Checkout Demo: https://merchantdemo.sandbox.bka.sh/

a. You can only use any of the below wallet numbers in sandbox environment. If one wallet gets locked/ inactive, please use the other one. PIN and OTP will be the same for all wallets.

<table>
<tr><th colspan="3">Wallet Numbers</th></tr>
<tr><td>Regular Active Customer wallet numbers:-</td><td colspan="2">1. 01770618575
2. 01929918378
3. 01770618576
4. 01877722345
5. 01619777282
6. 01619777283</td></tr>
<tr><td rowspan="2">Customer wallet numbers for Failed Cases:-</td><td>Insufficient Balance</td><td>01823074817</td></tr>
<tr><td>Debit Block</td><td>01823074818</td></tr>
</table>

b. **PIN**-12121

c. **OTP**-123456

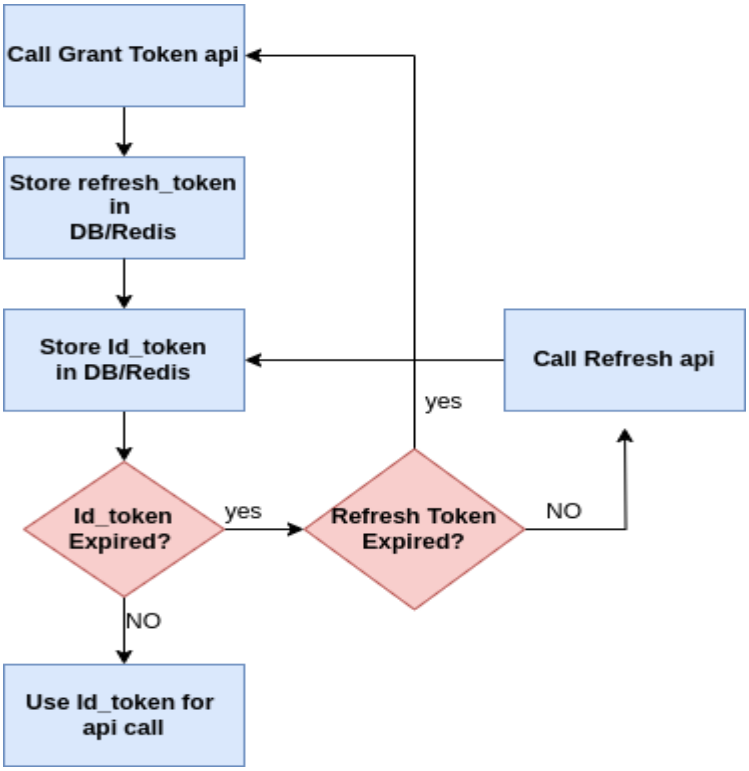
4. Github (https://github.com/bKash-HSL) for sample Code reference.

5. Payment Gateway error codes, subject to corresponding failure scenario (https://developer.bka.sh/docs/error-codes).

From integration perspective, we have to go through the following Milestones :

Milestones	Tasks
Step 1: Integration Initiation	Merchant is offered Developer Portal, Merchant Integration Portal, Demo Link and Sandbox credentials for testing
S-2: Merchant system readiness with bKash PGW Sandbox	a. Merchant confirms development readiness using sandbox
	B. Merchant shares user journey flow (if required)
	c. Solution document finalization (if required)
S-3: Sandbox result validation	Merchant Validate Create and Execute responses in Merchant Integration portal
S-4: Production info collection & Production onboarding	merchant is provided with live credentials in the Merchant Integration Portal and is onboarded for production.
S-5: Merchant system readiness with bKash PGW Production	Merchant confirms system readiness using Production credentials
S-6: UAT & Go Live	a. Technical UAT : Checking merchant backend security mechanism
	b. Business UAT : Checking user journey with Production credentials.
	c. Merchant makes the payment system available for all customers.

Token Management for bKash Integration



Overview

This document outlines the process of managing token expiration and refresh in a bKash payment gateway integration. It covers the initial token request, token usage, token expiration handling, and token refresh cycles.

1. Grant Token Request (Initial Call)

When a user first interacts with the bKash payment gateway, your application sends a request to the bKash {base_URL}/tokenized/checkout/token/grant endpoint to retrieve an id_token and a refresh_token. Store the tokens: After receiving the id_token and refresh_token, store them securely in your database or Redis. id_token: Valid for 1 hour. refresh_token: Valid for 28 days.

2. Check Token Expiry (Before Each Request)

Before each API call to bKash, check the expiration time of the id_token.

If the id_token is valid (not expired), use it for the API request. If the id_token is expired (after 1 hour), proceed to the next step.

3. Refresh Token Request

When the id_token expires, send a request to the bKash {base_URL}/tokenized/checkout/token/refresh API with the stored refresh_token as a parameter. Receive new tokens: The response will include a new id_token (valid for 1 hour). Update token storage: Replace the old id_token with the new one in your database/Redis. The refresh_token stays the same unless explicitly refreshed or revoked.

4. Handle Token Expiration and Refresh Token Expiry

Each time the id_token is refreshed, the validity of the refresh_token remains the same (28 days from when it was originally issued). If the refresh token expires or becomes invalid (after 28 days):

Call the /token/grant endpoint again to issue a new id_token and refresh_token. Replace both tokens in storage with the new values.

In addition,

0. You must reuse Id_token for 1 hour for any subsequent API request to bKash for any number of payment gateway users
1. For failure/cancel status from create payment API callback a "payment failed" message should be shown after redirection
2. Please check and confirm execute API is called only and only for success status from create payment API callback
3. For execute API statusCode 0000 is for success response rather than this for any other failure case you should display t the proper status message from the API response. You always get a response with statusMessage property
4. Query payment should only be called if there is no response from the execute API and inside that block of code. Other than that do not use this API in the API flow of a payment
5. A refund test will be performed during UAT. So please confirm if you are using the Refund API using same id token for refund transaction

**please confirm these action points as these are necessary for our technical acceptance. transaction".