Nagad Online Payment API Integration Guide

Version 3.3, October 2021

Disclaimer

Audience

This document is intended for the technical personnel of merchants and service providers that want to incorporate a new online payment method provided by Nagad.

Contents

List of Figures

Revision History

Version 1.3 – November 2019			**Change list**
Date	:	24-October-2019	• First Version
By Whom	:	Md. Sakib Anwar, Jr. Engineer Md. Badi- Uz-Zaman Shajib, Sr. Engineer	
Status	:	First Revision	
Reviewed By	:	Nazmul Quader Zinnuree	
Version 1.4 – October 2019			**Change list**
Date	:	24-October-2019	• Added Revision History • Added API Version • Added Sender Fee Sending Capability in Place Order API • Changed random to challenge in Initialize API and Place Order API
By Whom	:	Md. Sakib Anwar, Jr. Engineer	
Status	:	Second Revision	
Reviewed By	:	Md. Badi- Uz-Zaman Shajib, Sr. Engineer	
Version 3.0 – July 2020			**Change list**
Date	:	11-July-2020	• Add service fee in order placement (Section 5.2.8)
By Whom		Md. Sakib Anwar, Jr. Engineer	
Status		Third Revision	
Reviewed By		Md. Badi- Uz-Zaman Shajib, Sr. Engineer	
Version 3.1 – July 2020			**Change list**
Date	:	14-July-2020	• Added Purchase Status (5.2.7) • Updated Check Status API Response (5.2.8.3.3)
By Whom		Md. Sakib Anwar, Jr. Engineer	
Status		Fourth Revision	
Reviewed By		Md. Badi- Uz-Zaman Shajib, Sr. Engineer	
Version 3.2 – September 2020			**Change list**
Date	:	28-September-2020	• Added Convention for Logo URL and Locale in API Conventions and Constants 5.2.5 • Added locale as a request parameter for Initialize Request 5.2.8.1.2 • Added Sample Additional Merchant Info to be shown in Payment Pages Place Order Request 5.2.8.2.2
By Whom		Md. Sakib Anwar, Jr. Engineer	
Status		Fourth Revision	
Reviewed By		Md. Badi- Uz-Zaman Shajib, Sr. Engineer	
Version 3.3 – October 2021			

Date **By Whom** **Status** **Reviewed By**		Md. Badi- Uz-Zaman Shajib, Md Abdullah al Mamun	

1. Background

Aiming to lead the country's booming Digital Financial Service (DFS) revolution, Nagad is a venture by the Bangladesh Post Office that facilitates the day-to-day financial transaction needs of the people.

Nagad brand is operated by Third Wave Technologies Limited (TWTL).

The core philosophy of the service is to fuel possibilities. According to the World Bank's Global Findex 2017, only 50% of Bangladeshis had mobile banking and/or financial institution accounts as of 2017, of which only 21.2% have mobile money accounts. That leaves a large population of unbanked people who are in need of reliable service to assist their financial needs. With a host of essential services such as cash in/out, P2P money transfer and mobile top-ups, the purpose of Nagad is to enhance the lives of people by empowering them with financial flexibility.

Nagad sets itself apart by ensuring meticulous customer service. Through its facilities, the company contributes to the financial inclusivity & socio-economic development of the country.

Kona Software Lab Limited (the Bangladesh office of the South Korean Technology Pioneer KONA I Co., Ltd.) is the technology solution provider of Nagad.

2. Understanding Credentials

Merchant ID: This is a unique identifier provided to every merchant. The Merchant ID is part of the merchant's account credentials.

Order ID: This is unique identifier provided by merchant to place order for payment

Reference Id: This is unique identifier provided by e-commerce platform for payment. This will also can be used for future reference.

Merchant RSA Public-Private Key: Merchant will generate *RSA Key Pair*. The Private Key will be used to generate signature on merchant sensitive data before sending to Nagad Payment Gateway ("Nagad PG"). Merchant will upload public key to Nagad Online PG server via Portal. Nagad Online PG will use this key for encrypting sensitive data before sending response to merchant.

Nagad Online PG RSA Public Key: Nagad Online payment service will provide RSA public key which will be available in portal after merchant registration. Merchant will use this key to encrypt sensitive data before sending to Nagad Online PG. Merchant will verify signature with this public key which is sent in the response of Nagad Online PG APIs.

3. Onboard with Nagad Online PG

- **STEP 1—Register as a Merchant**
  - Merchant needs to be registered in the system with corresponding data through the Portal.
- **STEP 2—Get Nagad Online PG Credentials from Merchant Dashboard**
  - *Merchant ID* can be obtained from portal after successful completion of registration.
  - Nagad *Gateway RSA Public Key* can be obtained from portal.
- **STEP 3—Update Merchant Integration Credentials**
  - Generate Merchant RSA Key Pair (Merchant will generate this key pair).
  - Configure Merchant Public key in Portal (Upload Public Key through portal interface).
  - Merchant will store own private key securely for further communication (Merchant will save and keep this to use in future).
- **STEP 4—Integrate Nagad Online Payment API**
  - Nagad Online PG has several APIs which needs to be integrated by merchant for accessing different service provided by Nagad Online PG (e.g. payment, check payment status, etc.)

4. Nagad Online PG Call Flow

4.1 Payment

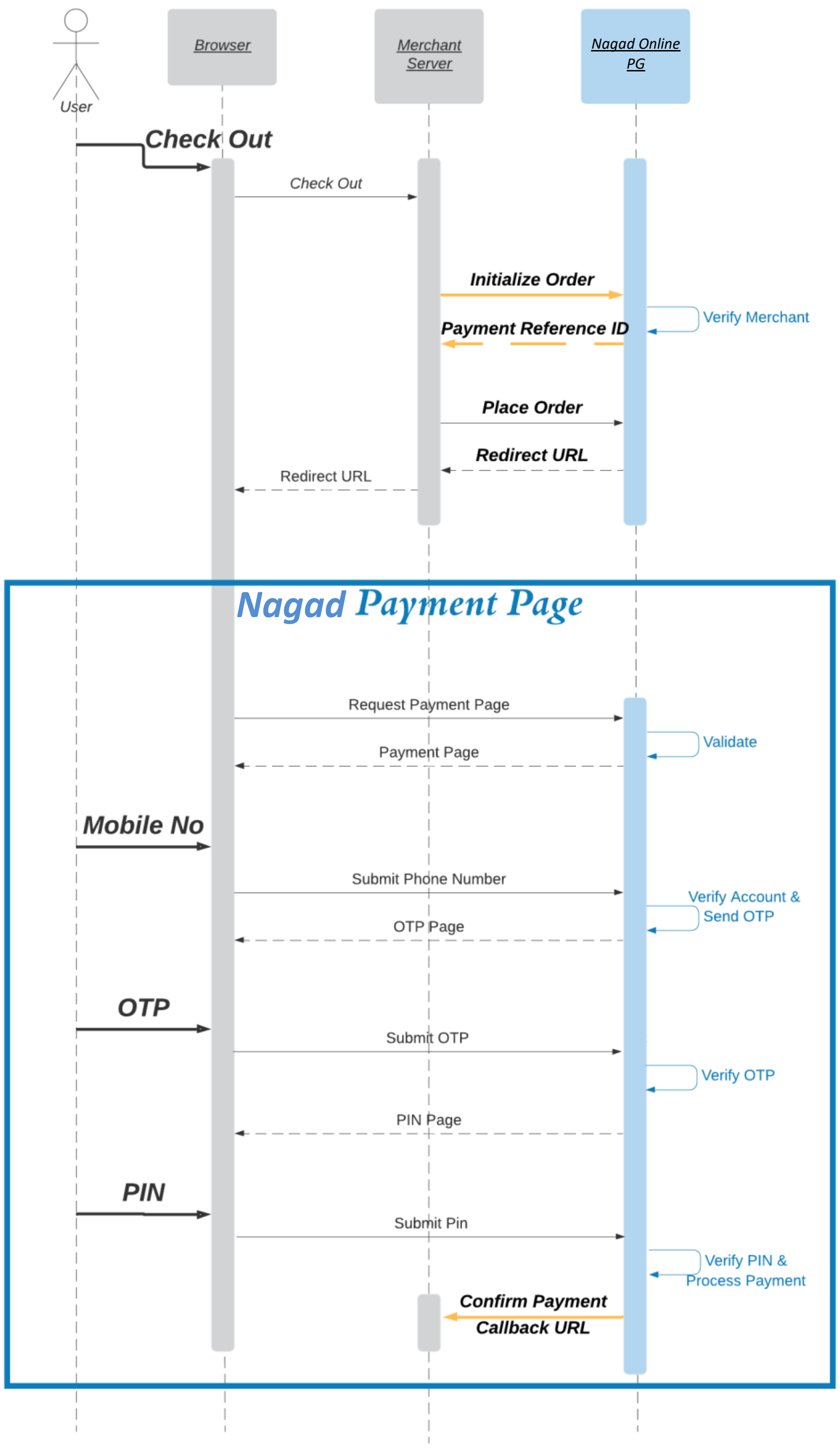


Figure 1: Payment Flow

4.2 Payment Status

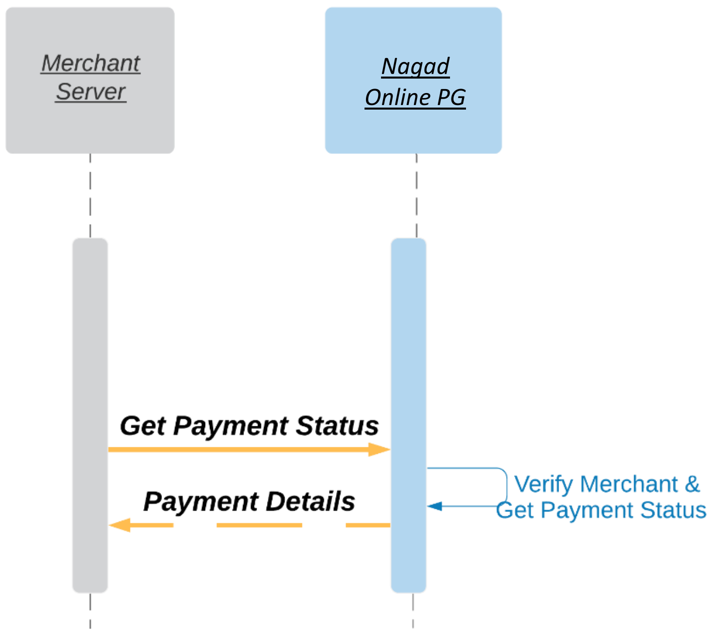


Figure 2: Payment Status

5. Integration with Nagad Online Payment API

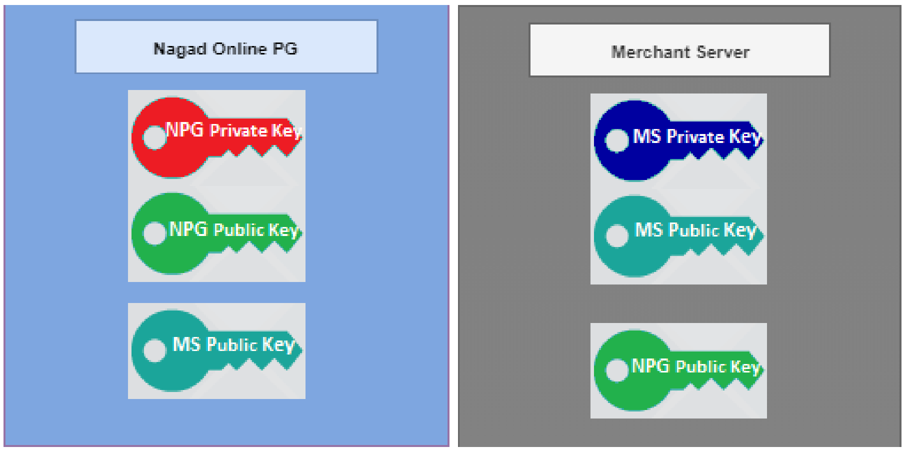
5.1 Merchant Registration and Configuration

After successful completion of merchant registration and enrollment for online payment service, merchant will have access to an integration console in portal. Merchant will get following information in the console:

- Merchant ID
- Nagad Gateway Public Key

Merchant will configure following data in the portal integration console

- Merchant server address
- Call back URL for payment confirmation (Conditional)
- Merchant Public Key (Mandatory)



5.1.1 Merchant Crypto Operation

Merchant need to perform following crypto operation for Request Data Preparation:

1. **Encryption**: Merchant need to encrypt the plain Sensitive Data (e.g. {"merchantId":"mer082738712637","datetime":"201910291828807","orderId":"ord0000001","random":"695EF3869547B6C07F5D56399935FB72D21737EA"}) using Nagad Gateway Public Key (e.g. NPG Public Key) with PKCS1Padding Algorithm.

```
Encrypted Sensitive Data: Encrypt(Plain Sensitive Data, NPG
Public Key, PKCS1Padding)
```

2. **Digital Signature**: Merchant need to generate signature using plain Sensitive data and Merchant Private Key (e.g. MS Private Key) with SHA1withRSA signature algorithm.

```
Generated Signature: Sign(Plain Sensitive Data, MS Private Key,
SHA1withRSA)
```

3. **Encoding**: Encode Encrypted Sensitive Data and Signature using Base64 Format.

```
sensitiveData: Base64_Encode(Encrypted Sensitive Data)
signature    : Base64_Encode(Generated Signature)
```

Merchant need to perform following crypto operation for Response Data Retrieve:

1. **Decoding**: Perform Base64 Decoding operation on received sensitiveData and signature

```
Decoded Sensitive Data: Base64_Decode(Received sensitiveData)
Decoded signature     : Base64_Decode(Received signature)
```

2. **Decryption**: Merchant need to decrypt the decoded Sensitive Data using Merchant Private Key (e.g. MS Private Key) with PKCS1Padding Algorithm.

```
Plain Sensitive Data: Decrypt(Decoded Sensitive Data, MS
Private Key, PKCS1Padding)
```

Decrypted data (Plain Sensitive Data) sample-
{"paymentReferenceId":"sample0nlinepaymentrefid","random":"40C88FFFF3274CD3698B140E7F7C211C415E0812","acceptDateTime":"20191029182852"}

3. **Signature Verification**: Merchant need to validate the Nagad Online Payment Gateway. Signature verification perform using Decrypted Sensitive Data, Decoded Signature and Nagad Gateway Public Key (e.g. NPG Public Key) with SHA1withRSA signature algorithm.

```
Verification Result: Verify(Plain Sensitive Data, Decoded
signature, NPG Public Key, SHA1withRSA)
```

5.2 Understanding Nagad Online Payment Gateway APIs

Nagad Online Payment Gateway follows secure HTTPS request response model. APIs are restful stateless. Safe secure communication is ensured by the strict encryption policy and request parameters.

5.2.1 HTTP Request URL

Every request URL is constructed in a particular fashion to ensure consistency across different component and easy understanding of the purpose of an API. General structure of an API is:

https://NAGAD-PAYMENT_BASE_URL:PORT/CONTEXT-PATH/API-PATH

- NAGAD-PAYMENT-BASE-URL is the URL provided to clients for communication
- PORT is the port of the component the client is trying to reach
- CONTEXT-PATH defines the component the client is trying to communicate with
- API-PATH is the path to a particular API

Example: https://sandbox.mynagad.com:12345/remote-payment-gateway-server-1.0/api/v2/init

5.2.2 HTTP Header

HTTP headers are used to pass some meta-data about the request and response which helps in identifying a request in its preliminary stage. The headers used in Nagad Payment Platform are:

- **X-KM-IP-V4**: This header is used to send the client IP e.g. "*10.55.247.69*"
- **X-KM-Client-Type**: This is a constant String which is used to identify client type. The allowed values for this header are:
  - PC_WEB
  - MOBILE_WEB
  - MOBILE_APP
  - WALLET_WEB_VIEW
  - BILL_KEY
- **X-KM-Api-Version**: The version of Nagad Platform the client is willing to use e.g. "v-0.2.0".
- **Content-Type**: application/json for POST and PUT method. In response header content type is always application/json.

5.2.3 HTTP Request Body

The request body depends on the type of request made. APIs use method that is most applicable for that particular API and its purpose.

5.2.3.1 GET Request

GET request can be made with either path variables or request parameters.

- Path variables are a part of the API that is being called.
  For example: HTTP://SANDBOX.MYNAGAD.COM:PORT/CONTEXT-PATH/api/init-order/{reference-id}
  Here, reference-id is a variable that needs to be filled in with actual value while calling the API.
- Request Parameters are appended with the URL at the end of the URL.
  For Example: HTTP://SANDBOX.MYNAGAD.COM:PORT/CONTEXT-PATH/API-PATH?param1=value1&param2=value2
  Here, param1 and param2 are two parameters and value1 and value2 are the values associated with them

5.2.3.2 POST Request

A POST request usually contains a body in JSON structure. The JSON must contain some key and values associated with them for the API to work properly. Request Parameters can also be used to pass data to API via POST method if needed.

For Example: A request body may contain a JSON like below:

Example

```
{
  "key1": "value1",
  "key2": {
    "key3": [
      "value2",
      "value3"
    ]
}
```

Here there are two keys with another array inside one of them. Nested structure such as this are complicated to understand and will not be used in most cases. However, this is the structure that is being followed by the APIs we are providing.

5.2.4 HTTP Response Body

Depending on the API the response body might vary. However, most of the API responses are based on the calling methods. GET calls usually returns an information and thus varies from API to API. Other methods such as POST, PUT, and DELETE calls follow a particular structure:

1. **Response Message**: This contains a brief message from API and depends on the result of the request made. For example, API usually sends success message if a request is successfully processed by it.

2. **Error Code**: Error codes are provided when the API runs into an error. The error codes are made up of three individual parts joined by '_' that makes it easy to understand the reason behind the failure of the request made.
3. **Others**: According to the need of every API they may send some extra information that in expected in response to further continue the payment process.

Example

```
{
   "reason" : "16_0006_001",
   "message" : "Invalid Pin Provided"
}
```

5.2.4.1 HTTP Status

The HTTP statuses follow the below given structure:

Status	Response
2xx	Successfully Processed
4xx	Error occurred in client's part
5xx	Error occurred in server's part

5.2.5 API Conventions and Constants

1. The context path for Nagad Platform is **remote-payment-gateway-1.0**.
2. Date format

Format	Meaning
yyyy	Year in 4 length e.g. 2014.
MM	Month in 2 digit e.g. For October value is 10.
dd	Day of the month. e.g. 21 for 21th of any month.
HH	Hour in 24 e.g. 1:00 pm is 13.
mm	Minute of the hour e.g. 12.
SS	Second of the minute.

3. M/O/C column stands for

M/O/C	Meaning
M	Mandatory
O	Optional
C	Mandatory based on condition

4. Logo must be in the size of 200 px x 200 px. The width can be less than 200 pixels depending of the aspect ratio of the logo.
5. Supported Locale where applicable are **EN** for English and **BN** for Bangla. By default **EN** will be chosen.

5.2.6 API Version

API Version related to both the SDK the client is using and the feature associated with it. Currently there are two version.

1. **v-0.2.0**
   This is the base version and offers all the basic features such as transaction and status check.
2. **v-3.0.1**
   This version provides the capability of providing Sender Fee along with principal amount while placing the order. The user will be charged for both the principal amount and the sender fee.

5.2.7 Purchase Status

Purchase Status defined the state a transaction is in. Operations allowed on a particular transaction request depends on the state it is in. Some of the states are end state and some of them are transient ones kept to maintain the life cycle of a transaction. Current Purchase Statuses are:

1. Success
2. OrderInitiated
3. Ready
4. InProgress
5. OtpSent
6. OtpVerified
7. PinGiven
8. Cancelled
9. PartialCancelled
10. InvalidRequest
11. Fraud
12. Aborted
13. UnKnownFailed
14. Failed

5.2.8 Integration APIs

5.2.8.1 Initialize

The communication channel and session will be initialized using this API. This API generates a payment reference number (Payment Ref Id) to communicate with merchant in further operations to successfully complete a payment using Nagad Platform.

5.2.8.1.1 Header

Defined in previous section HTTP Header.

5.2.8.1.2 Request Method & Path

Method	URL
POST	/remote-payment-gateway-1.0/api/dfs/check-out/initialize/{merchantId}/{orderId}?locale={locale}

5.2.8.1.3 Parameters

Type	Name	Data Type	MOC	Length	Description
Path Variable	merchantId	String	M	15	ID provided to Merchant after registration
	orderId	Alpha Numeric String	M	5~20	Unique Order Id generated by merchant to identify a particular request
Request Param	locale	String	O	2	Default locale for payment page according to API Conventions and Constants
Request Body	accountNumber	Number String	O	11	Merchant Mobile Number (Applicable if merchant has an account with Nagad)
	dateTime	Numeric String	M	14	Date and Time of Payment in the format (yyyyMMddHHmmSS). API Conventions and Constants
	sensitiveData	Base64 Encoded String	M	1~1024	Base64 encoded encrypted data.
	signature	Base64 Encoded String	M	1~1024	Base64 encoded merchant signature.

5.2.8.1.4 Example

Request
POST *http://sandbox.mynagad.com:10080/remote-payment-gateway-1.0/api/dfs/check-out/initialize/mer082738712637/ord0000001*   {"**accountNumber**":"01701892123","**dateTime**":"201910291828807","**sensitiveData**":"G58zmiHNIT+CM74fQyL6+w0WdPXioW6oZy1piRABi1ssj3vt89LZoLPh/0f2J/y5vziNQHW4GfZ2dfvY/PC2v5BOD0njDtxLewXenRyO0+nbtbJRDjl8RspmWxTG3y4aO0Pxjlr+B/h0cypj1AoGaHEY5UcqyVHyGW0C5sYK+8GJV7IV1PwP9CcnwkVnbBB4TMN098HMn+3BJd7qJp/YdyUfZSkvdDQyqPVobjH4IbhyJH4PirwCrxoDneUdV2WH57RlX786S8/yQ7Tvvl6Y3rS6RWyWAUnfJeKFRSPIjGGTibKlFf9Imm6hflvA0H5za4SZHqo4KUcyATISk149fg==","**signature**":"AlhsvF6ZdEUbDqXeeHeS6Ab9e+/W8U4pxZjMr5+qx5aGIDj21R6qGiYFiHm9JyH

+vYs2+VBzIZVhhaV7P3WCXI8fOxz0ThYdFgN3139kHj9gcRWIM5A82gnwjWmwKiOMdLWM/PVy6jJgF
UiMXYz5z7JFTNbEUk5tS4uXAjbUPsqpXMKYY+CZZQYqYDguo+yDh/gepOL/tKLbp40SpjTtdmOG2elJxr
BRvvyFNMIsrDb3DzS+wGrBUd9mMBeioL5CYZ4GFNHvVKm7B2+w5rQ17ad1p6YmKjq/9HI0/101odxI2
6EwK7PifYKnMGj/X6bPKk2GeJ5RR0gbsTK5X6fz1w=="**}**

5.2.8.1.5 Data elements for *sensitiveData*

***sensitiveData* Fields**	**Data Type**	**Length**	**Description**
merchantId	String	15	ID provided to Merchant after registration
dateTime	String	14	Date and Time of Payment in the format (yyyyMMddHHmmSS). API Conventions and Constants
orderId	Alpha Numeric String	5~20	Unique Order Id generated by merchant to identify a particular request
challenge	Hex String	40	Random number that should be generated by merchant.
Example			
{"**merchantId**":"mer082738712637","**dateTime**":"201910291828807","**orderId**":"ord0000001","**challenge** ":"695EF3869547B6C07F5D56399935FB72D21737EA"**}**			

5.2.8.1.6 Response

Type	**Name**	**Data Type**	**MOC**	**Length**	**Description**
Response Body	sensitiveData	Base64 Encoded String	M	1~1024	Unique Id provided for identification of a particular request in further stages
	signature	Base64 Encoded String	M	1~1024	Sensitive data are encrypted and sent as string which can later be decrypted by client and used

5.2.8.1.7 Example

Response
{"**sensitiveData**":"C5aETMhx5UexvlO0fSNN9YFWwCSJrRO4PjHCW+yHw0pkVtQvRf11Rg4iIucbj2Tfz4lBD+PUyFRQ/FO/9/5seBcPxOhEYzXeCky9C/9FEjCYrGGZuG300fgIc3IQzzu80hcLQmoafeqkOiwRUqBa9w+fSFDPbmIKQHSpPPR5aP44L1f12PNvxXH2mWJrheFcLNu+vDjaWrnKmAVAYF0o5AXUfBeI+Cs7MP8mE9mLsbm7GNR26BACPib9SIsmxpnq0TZAsJxCvunDzmgto8P7TMTZ8KqVfGqTu97Kp4M6rxMabUV5YZVSUq1aj84f9qObWsUNRP3Xb6wPwNEB2qR3DA==","**signature**":"hD2RD8ZKUhBjKTnwnvS+pF2vWrOTTOfw1z1G7U6WYYPBM4SShzPeKUvUCBKAvZu7SHRzAYu334khg3n2HoJknNCuFzqh5xAKjcitgXJlFU3sFiE4nmFRcoLcVqLq+5lS+nNgM72ImIAEqVPViCfRskr3gV9iXO5WDnWeJlfs85kLL2ND/MLphzZKY14R9M6TY20z07ZCLBIcz1wRc3qNgDTEIS76QN61pWalELdT74ISyyQC5CwjS+p4ZkF2HfIZfWJ4HrqdHPQRZ0osIVgGV81GZlNmd2qaFblWLKzuAq+jL3Go4QiaCgn1pO/2d9ZlfrNUwBiFFoWSBitB25XQKA=="**}**

5.2.8.1.8 Data elements for *sensitiveData*

***sensitiveData* Fields**	**Data Type**	**Length**	**Description**
paymentReferenceId	String	10~60	ID provided to Merchant for further transaction
acceptDateTime	String	14	Date and Time of Payment recieved in the format (yyyyMMddHHmmSS). API Conventions and Constants
random	Hex String	40	Random number that should be generated by merchant.
Example			
{"**paymentReferenceId**":"sample0nlinepaymentrefid","**random**":"40C88FFFF3274CD3698B140E7F7C211C415E0812","**acceptDateTime**":"20191029182852"}			

5.2.8.2 Place Order

The next step of completing an order is sending the amount and other payment information. The purpose is to establish a secure communication channel and passing all the order related information to Nagad Platform.

5.2.8.2.1 Header

Defined in previous section HTTP Header.

5.2.8.2.2 Request

Method	**URL**
POST	/api/dfs/check-out/complete/{paymentRefId }

5.2.8.2.3 Parameters

Type	**Name**	**Data Type**	**MOC**	**Length**	**Description**
Path Variable	paymentRefId	String	M	10~60	ID provided to Merchant after initialization
Request Body	sensitiveData	Base64 Encoded String	M	1~1024	Encrypted Sensitive Data
	signature	Base64 Encoded String	M	1~1024	Signature Generated by Merchant Private Key

	merchantCallbackURL	String	C	10~255	Callback url for getting transaction result
	additionalMerchantInfo	Map<String,String>	O	1~2048	Additional Merchant Data

Example

```
{
   "sensitiveData":
"PSFButymlhAlKrOjiG+RKrz4uETizC9Z0mueKMRvqi62Ctz+o4AQ3+8Z/08AEs1Q215u6+fgA4
OcussnCXH2W0ghuF9p5n0uR8waLYE8llLaUcQkAGnfBbUOzvvtCyZdfu4dtTnbQt0jJvs8m7eV
o6xoKjjhGsXIdhOXML6kR8MQPQjBCkCtgLBxX2MMmb5eo0IDULZkqZi+A9FRAM/OcwPI0ip
kFcnbDqwrjfts+ZFK7+FXOgy6ZntAYOlLfeD+O7m/jdPWm73upX8WMVSNM/HcCB5Au38Zg
+5kMzghIWXaNYtyIYxeZN0d0+fdrqBHe7Q4u0xnhzyG4DT9PhfOKQ==",
   "signature":
"hANZKhCwPZEbP5brZ6Nh9JnOgcrkBdOSnznPN0Mk5vS0rs3Ta/gPeCZH2XBBnN6emkdRqU
QCoGFtoN8GMBmfjbqf04di3hggGs0n0LrK6dr25QnqIJ9qKmylMFRHPouZ24tDf7i7rHwLrJF7r
5l9NxPqOdomBPHwNeJW/z2snIwNbEYmjK/YwtjPRSVxTBgBz7OzamVfmXERmuUifXa26uqX
D/jKJzJ35LxPGRcgdWC/c0LcfDBXiYr7lFO9PsMB33HUDQWQQzAxc8pl77HHO9jVObD11JJ0u
i/GnhJZlvAyHcKlNuxY+Kkwod1bMYgkZ32dtkB5O1I5xDY03OTcuw==",
   "merchantCallbackURL": "http://merchant.kpp.com/payment-result",
   "additionalMerchantInfo": {
         "productName":"shirt",
         "productCount":1
   }
}
```

Sensitive Data Field	Data Type	Length	Description
merchantId	String	15	Same as Initialization
orderId	Alpha numeric stirng	5~20	Same as Initialization
amount	String	15	Amount of Payment up to two decimal point e.g. 1722.83.
currencyCode	String	3	Currency Code in which payment is being made e.g. "050" for BDT if not present default is "050"
challenge	String	40	Random number found from Initialization response
otherAmount	JSON Object		Other Amount related to transaction e.g. Sender Fee, Charge etc.

Example

```
{
   "merchantId": "mer082738712637",
   "orderId": "ord0000001",
   "currencyCode": "050",
   "amount": "1790.00",
   "challenge": "40C88FFFF3274CD3698B140E7F7C211C415E0812",
   “otherAmount” : {
       “serviceFee” : “2.56”
    }
}
```

Other Amount Field	Data Type	Length	Description
serviceFee	String	15	Amount of Sender Fee up to two decimal point e.g. 22.83. Only allowed for some API version.
Example			
{ “serviceFee” : “2.56” }			

Additional Merchant Info Fields	Data Type	Length	Description
serviceName	String	25	Service Name Provided by Merchant
serviceLogoURL	String	1~1024	Publicly accessible logo URL. Logo must abide by the specifications mentioned in API Conventions and Constants
additionalFieldNameEN	Alphanumeric String	20	Additional Field Name to be shown in Payment Page for Locale EN
additionalFieldNameBN	Alphanumeric String	20	Additional Field Name to be shown in Payment Page for Locale BN
additionalFieldValue	Alphanumeric String	20	Value of Additional Field in **English**
Example			
{ “serviceName” : “T Shirt”, “serviceLogoURL” : “tinyurl.com/sampleLogoUrl”, “additionalFieldNameEN” : “Color”, “additionalFieldNameBN” : “রং”, “additionalFieldValue” : “White” }			

N.B: Additional Merchant Info can be anything and will be saved for further usage. However only these fields will be shown in the payment page.

5.2.8.2.4 Response

Type	Name	Data Type	MOC	Length	Description

Response Body	callBackUrl	String	M	1~1024	Redirect url to Nagad PG page.

5.2.8.2.5 Example

Type	Name	Value	Description
Redirect URL to Nagad PG Page.			
Example			
{ "callBackUrl": "https://URL/payment/aser834sdafdsf0awerasdfasdr5" }			

5.2.8.2.6 Payment Success Callback – Payment Result

Type	Name	Data Type	MOC	Length	Description
Request Param Variable	merchant	String	M	15	ID provided to Merchant after initialization
	order_id	Alpha numeric stirng	M	5~20	Parameters
	payment_ref_id	String	M	10~60	Parameters
	status	Enum	M	2-20	Success, OrderInitiated, Ready, InProgress, Cancelled, InvalidRequest Fraud, Aborted, UnknownFailed
	status_code	Map<String, String>	M	1~20	Additional Merchant Data
	payment_dt	String	M	14	Date and Time of Payment in the format (yyyyMMddHHmmSS). API Conventions and Constants
	issuer_payment_ref	String	M	8~20	Reference from issuer for transaction reference
Example					

```
http://MERCHANT-IP:PORT/payment-
result?merchant=1236545412342534&order_id=ordasdf234&payment_ref_id=sample0nl
inepaymentrefid&status=success&status_code=00_000_000&payment_dt=20192906162
245&issuer_payment_ref=70X8BCDI
```

5.2.8.3 Check Payment Status

This API returns the status of a payment that took place earlier. If the call back URL provided during merchant registration is unable to handle the response from Payment Gateway, then it can manually check the status of a payment via this API.

5.2.8.3.1 Header

Defined in previous section HTTP Header.

5.2.8.3.2 Request

Method	URL
GET	/api/dfs/verify/payment/{paymentRefId}

5.2.8.3.3 Response

Type	Name	Data Type	MOC	Length	Description
Response Body	merchantId	String	M	15	ID provided to Merchant after registration
	orderId	Alpha numeric stirng	M	5~20	Unique Order Id generated by merchant to identify a particular request
	paymentRefId	String	M	8~60	ID provided to Merchant after initialization
	amount	String	M	1~15	Amount of Payment up to two decimal point e.g. 1722.83.
	clientMobileNo	Numeric String	O	11	Payee's mobile number.
	merchantMobileNo	Numeric String	M	11	Mechant Mobile Number
	orderDateTime	String	M	14	Date and Time of Payment in the format (yyyyMMddHHmmSS). API Conventions and Constants

	issuerPaymentDateTime	String	O	14	Date and Time of Payment in the format (yyyyMMddHHmmSS). API Conventions and Constants
	issuerPaymentReferenceNo	String	O	8~20	Reference from issuer for transaction reference
	additionalMerchantInfo	Map<String,String>	O	1~2048	Additional Merchant Data
	status	Enum	M	5~50	Purchase Status
	statusCode	String	O	1~20	Result code
	cancelIssuerDateTime	String	O	14	Date and Time of Payment in the format (yyyyMMddHHmmSS). API Conventions and Constants
	cancelIssuerRefNo	String	O	8~20	Reference from issuer for transaction reference

5.2.8.3.4 Example

Request	**Response**

GET **https://URL:PORT/remote-payment-gateway-1.0/api//dfs/verify/payment/trx1234567890**	{
	"merchantId": "687450000031324",
	"orderId": "NAG15731239931573",
	"paymentRefId": "MTEwNzE2NTMxODgwNC42ODc0NTAwMDAwMzEzMjQuTkFHMTU3MzEyMzk5MzE1NzMuZDA3Mjg5YTQxNDRhNWVjYzcxYjU=",
	"amount": "130",
	"clientMobileNo": null,
	"merchantMobileNo": "01745000003",
	"orderDateTime": null,
	"issuerPaymentDateTime": "20191107165357",
	"issuerPaymentRefNo": "70IZV6G7",
	"additionalMerchantInfo": null,
	"status": "Success",
	"statusCode": "00_000_00"
	}

5.2.9 Error Response Format and Error Codes

5.2.9.1 Error Response Format

Type	Name	Data Type	MOC	Length	Description
Response Body	reason	String	M	11~15	Error code
	message	String	M	1~1000	Error message

5.2.9.2 Example

Type	Name	Value	Description
Example			

```
{
 "reason": "16_0006_004",
 "message": "Provided merchant ID is invalid"
}
```

5.2.9.3 Error Codes

Code	Message
16_0006_004	Provided merchant ID is invalid
16_0006_052	Invalid Merchant
16_0006_053	Inactive Merchant
16_0006_056	Encryption failed
16_0006_057	Decryption failed
16_0006_058	Failed to verify signature
16_0006_059	Invalid Sensitive Data
16_0006_060	Error processing sensitive data
16_0006_061	Invalid merchant key
16_0006_064	Mandatory Header Missing
16_0006_068	Invalid Order Id
16_0006_076	Transaction Date Time Not in allowed window
16_0006_075	Could not persist data to storage
16_0006_081	Invalid Date Time Format
16_0006_083	Duplicate Order ID in same day
16_0006_999	Invalid Request
16_0006_017	Purchase information state is invalid
16_0006_040	Invalid encrypted request type
16_0006_050	Provided merchant ID is invalid
16_0006_055	Invalid Payment Reference Id
16_0006_069	Data not encoded
16_0006_080	Invalid Currency Code

6. Typical Payment Flow with Nagad Online Payment API

Payment flow will be initiated when a customer wants to purchase some products from an online e-commerce platform and is ready to check out. Payment options will include *Pay with Nagad* which takes customer to the payment flow.

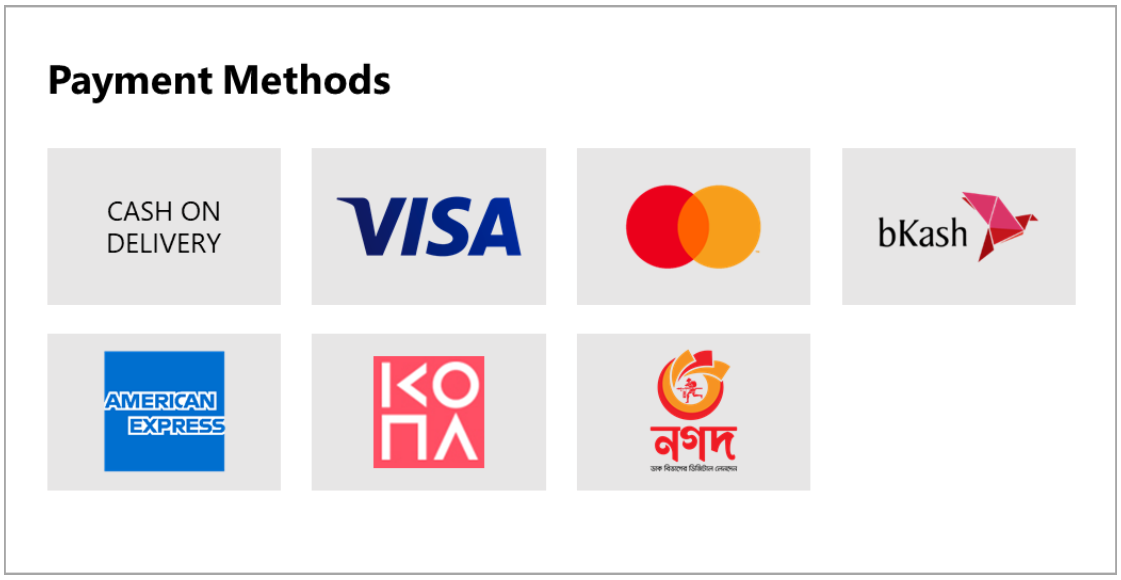


Figure 3: Nagad as a Payment Option on e-commerce checkout page

- ➔ Customer clicks on the payment icon.
- ➔ Customer will be redirected to Nagad Payment page i.e. *Link URL*.

The merchant website takes control.

- ➔ Link URL page displays

  - *Merchant Name*
  - *Order ID*
  - *Payable Amount*
  - *Input: Customer Account Number*
- ➔ Customer enters Account Number and will press Next.
- ➔ Nagad System generates OTP and sends it to the customer via SMS.
- ➔ Customer enters OTP on the next page.
- ➔ System verifies OTP and proceeds to PIN verification.
- ➔ Customer enters PIN to authorize payment.
- ➔ Nagad processes the payment and redirects to the Merchant Callback URL with transaction-ID.

The merchant website takes control.

- ➔ Merchant received the request from Nagad Platform and verify the transaction with Nagad Platform.
- ➔ Merchant server then proceeds to order processing internally.

Kona Software Lab Limited

Police Plaza Concord, Tower-A, 8th Floor, Unit-H, Plot-2, Road-144, Gulshan-1, Dhaka-1212, Bangladesh.
Tel: +880-2-55045191 | Fax: +880-2-55045192
E: KSLTM@konasl.com

KONA I Co., Ltd. (Headquarters)

8F, 3, Eunhaeng-ro, Yeongdeungpo-gu, Seoul, Republic of Korea.
Tel: +82-2-2168-7500 | Fax: +82-2-769-1670
E: globalsales@konai.com

Third Wave Technologies Limited

Delta Dahlia Tower (Level 13 and 14), 36 Kemal Ataturk Avenue, Banani, Dhaka -1213
Tel: 096 096 16167
E: info@nagad.com.bd