Gestion des callbacks et notification de paiement

Dans cette section, vous trouverez tout ce qu'il faut pour gérer les différents points d'entrée que votre application recevra pendant et après un paiement Axepta BNP Paribas Online.

Vous apprendrez :

- quelles URLs transmettre lors de la création d'un paiement,
- dans quel ordre elles sont appelées,
- comment interpréter les informations retournées,
- comment vérifier le statut réel d'une transaction,
- comment gérer le webhook

Table des matières

1. Déclarer les URLs : return, cancel, webhook

Lors de l'initialisation du paiement, vous devez fournir un objet urls :

Exemple d'url de retour

```
        urls{
                "return":"https://myProcessingServer.net/myApi/success.php?transId=95330876-67ae-4949-a11c-
b9a29257831b",
                "cancel":"https://myProcessingServer.net/myApi/cancel.php?transId=95330876-67ae-4949-a11c-
b9a29257831b",
                 "webhook":"https://myBackOfficeServer.net/webhook.php"
        }
```

- return = redirection lorsque le client clic sur le bouton de validation à la fin du paiement.
- cancel = redirection si le client annule
- webhook = notification asynchrone et garantie, indépendamment de l'action du client

2. Return URL & Cancel URL

Une de ces deux url (return ou cancel) sera appelée à la fin du traitement de la transaction, afin de:

- **Rediriger le client sur le site marchant** a la fin de la transaction
- Informer le commerçant du **statut de la transaction** : validée ou non.
- de retourner au back-office du commerçant l'identifiant unique de la transaction **payId** générée par Axepta Online.

À l'appel de l'une de ces URLs, Axepta BNP Paribas Online ajoute automatiquement le paramètre : PayId=<paymentId généré par Axepta> - cf Préconisation d'intégration

Ajoutez un identifiant propre à votre système ans vos URLs afin de relier le retour à votre commande.

Exemple d'url appelé lors de la validation du paiement par le client.

- en vert, l'url passée lors de l'initialisation du paiement, avec le transId ajouté pour lier le retour à un id marchand.
- en rouge, le paramètre PayId ajouté par le serveur.

https://myProcessingServer.net/myApi/success.php?transId=95330876-67ae-4949-a11c-b9a29257831b&PayId=b6eae9b16e3343fa90da39d4ee7bf4ad

Que faire lorsque vous recevez cet appel ?

À l'appel de l'une de ces URLs :

1. Récupérez le paramètre PayId.
2. Appelez l'API pour obtenir l'état réel de la transaction : GET /payments/getByPayId/{payId} - Retrieve payment details by Payment ID
3. Mettez à jour votre commande en fonction du status et du responseCode.

Champs importants dans la réponse API

- Montant : value, capturedValue, refundedValue
- Identifiants : payId, transId, xId, refNr
- Statut : status = AUTHORIZED, CAPTURE_REQUEST, OK, FAILED, etc.
- Résultat :
  - responseCode = "00000000" OU "0" si succès
  - responseDescription = message textuel

Lorsque la transaction aboutie avec succès, le code réponse peut être "00000000" ou "0", suivant l'étape de la transaction, ou suivant le moyen de paiement utilisé.

- Pour savoir si le résultat est un succès, vous devez tester les 2 valeurs, "00000000" OU "0"

Exemples de réponses résumées

Réponse getByPayId pour une transaction par carte au statut AUTHORIZED


```
{
	"amount":{
		"value":126,
		"currency":"EUR",
		"capturedValue":0,
		"refundedValue":0
	},
	"payId":"91a6299a704147bf934aabd79fd1dc5d",
	"merchantId":"MY_MERCHANT_ID",
	"transId":"Trans361039",
	"xId":"b55e68b7e4644a90836ae31effe1fc60",
	"refNr":"refNb77254",
	"status":"AUTHORIZED",
	"responseCode":"00000000",
	"responseDescription":"Transaction successful",
	"paymentMethods":{
		"type":"CARD"
	}
}
```

Réponse getByPayId pour une transaction par carte au statut CAPTURE_REQUEST


```
{
	"amount":{
		"value":1200,
		"currency":"EUR",
		"capturedValue":0,
		"refundedValue":0
	},
	"payId":"09526745fa704e9c8584dbe893c31f99",
	"merchantId":"MY_MERCHANT_ID",
	"transId":"1230861007",
	"xId":"2781d7d379e5449e9717c901ba6f9ff7",
	"refNr":"Q1ovVxioaZ6n",
	"status":"CAPTURE_REQUEST",
	"responseCode":"0",
	"responseDescription":"REQUEST",
	"paymentMethods":{
		"type":"CARD"
	}
}
```


3. Webhook

A la fin du traitement de la transaction, Axepta BNP Paribas Online notifie le site marchand du résultat final de la transaction.

La notification Webhook est le seul moyen fiable d'être informé de la complétion d'une transaction. Il est impératif pour le site marchant de traiter les requêtes reçues sur l'Url Webhook.
Elle est réalisée par un appel http REST sur l'url webhook fournis lors de l'initialisation du paiement.

Elle est envoyée même si le client :

- ferme son navigateur,
- perd la connexion,
- ne revient pas sur votre site.

Quand est-il envoyé ?

À chaque fin de traitement de paiement asynchrone.

Si l'adresse appelée n'est pas accessible, la notification est répétée jusqu'à 8 fois

Essai	Délais	Temps après 1ère notification
0	instantané	0
1	00:01 h	00:01 h
2	00:08 h	00:09 h
3	00:27 h	00:36 h
4	01:04 h	01:40 h
5	02:05 h	03:45 h
6	03:36 h	07:21 h
7	05:43 h	13:04 h
8	08:32 h	21:36 h

Ce que vous devez faire

1. Lire la payload JSON
2. Identifier la transaction via payId ou transId
3. Mettre à jour votre système
4. Répondre un code 200 OK

> Important : ne déclenchez jamais la finalisation de commande uniquement à partir du Return URL.
>
> Utilisez toujours le webhook comme source d'information fiable.

Structure du webhook

Champs fournis

- merchantId
- payId
- transId
- xId
- refNr
- status (AUTHORIZED, FAILED, etc.)
- responseCode / responseDescription
- amount
- paymentMethods
- creationDate (UTC)
- channel (ECOM, MOTO, Pay By Link…)

Exemple

webhook payload

```
{
    "merchantId": "YOUR_MERCHANT_ID",
    "payId": "91a6299a704147bf934aabd79fd1dc5d",
    "transId": "Trans361039",
    "xid": "b55e68b7e4644a90836ae31effe1fc60",
    "refNr": "refNb77254",
    "status": "AUTHORIZED",
    "responseCode": "00000000",
    "responseDescription": "Transaction successful",
    "amount": {
        "value": 126,
        "currency": "EUR"
    },
    "paymentMethods": {
        "type": "CARD"
    },
    "creationDate": "2025-10-30T11:27:57Z",
    "channel": "ECOM"
}
```

Sécurité et Vérification

Afin de garantir l'authenticité des données du webhook, elle sont signées avec un HMAC-SHA256.

La signature est portée par 3 header http dans le message webhook.

Signature headers

X-Paygate-Signature-Version	Version du format de la signature (valeur fixe, actuellement, `v1`)

X-Paygate-Timestamp	Unix epoch timestamp (secondes depuis 1970-01-01T00:00:00Z, UTC)
X-Paygate-Signature	Signature dans le format v1=<hex-hmac> Peut supporter de multiple signature, pour gérer le renouvellement des clés (exemple : v1=<hmac1> OR v2=<hmac2>)

Multiple Clés

Il est possible de configurer jusqu'à 2 jeux de clés, pour l'Authentification API , et pour la génération des Signature.

Les clés sont définies dans le portail marchant, et sont identifiées comme:

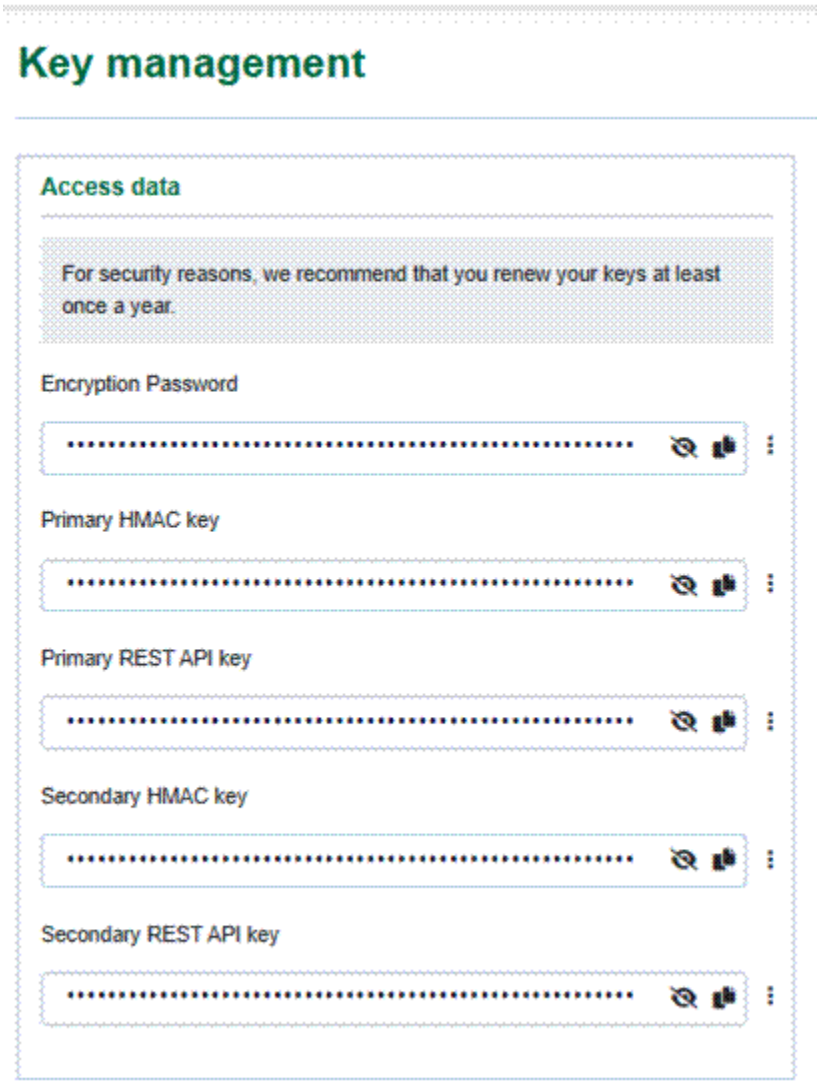
- Primaire (v1)
- Secondaire (v2)

La version de la clé utilisée par la plateforme Axepta Online pour signer les webhook dépend de la version de la clé utilisée par le commerçant lors de l'authentification.
Par exemple, si le marchant s'authentifie avec le jeu de clé REST API Secondaire (v2), alors la clé REST HMAC Secondaire (v2) sera utilisée pour la signature, et l'entete http X-Paygate-Signature commencera par : "v2=...".

La version de la clé utilisée par la plateforme Axepta Online pour la signature des webhook est la même que celle utilisée par le Marchant lors de du processus d'authentification.

Rotation des clés

Deux jeux de clés peuvent etre configurés, mais un seul est actif à la fois (celui choisi lors de l'authentification).
Ceci permet de mettre a jour le jeu de clé inactif, de le configurer et le déployer sur l'ensemble des équipements avant de l'activer.



Génération de la signature (Axepta BNP-Paribas Online)

```
signed_payload = timestamp + "." + raw_json_body
signature = HMAC_SHA256(secret, signed_payload)
```

- `secret` – HMAC key
- La `signature` générée est encodé 'hex' et le header est valorisé comme suit:

```
X-Paygate-Signature: v1=<hex-hmac>
```

Vérification de la signature (Commerçant)

Pour vérifier l'authenticité des données du message webhook:

1. Extraire les données portées par les headers HTTP:
    - `X-Paygate-Timestamp`
    - `X-Paygate-Signature`
2. Prendre les données brutes de la payoad Json (les données binaire reçues)
3. Calculer le HMAC avec les données extraites
4. `signed_payload = timestamp + "." + raw_body`
   `expected_signature = HMAC_SHA256(secret, signed_payload)`

5. Comparer la signature calculée avec celle reçue en utilisant une méthode "constant-time"
6. Valider le webhook si
    a. Les signatures sont identiques
    b. le Timestamp est compris dans les ±5 minutes par rapport à votre horloge locale.

Recommandations d’intégration

- Toujours vérifier la transaction via le endpoint getByPayId, même si le retour client laisse penser à un succès.
- Toujours traiter le webhook.
- Toujours vérifier l'authenticité des données du webhook en validant sa signature.
- Logger tous les appels return/cancel/webhook.