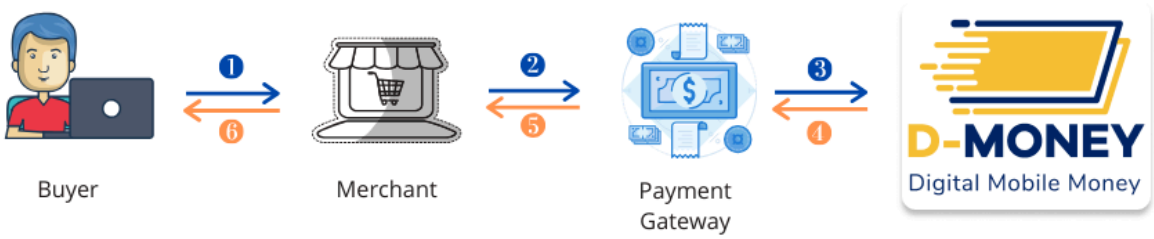

PAYMENT GATEWAY API DOCUMENTATION

Il s'agit d'une documentation expliquant comment se connecter aux systèmes de D-Money.

Name	D-Money Payment Gateway API Documentation
Version de Document	2.1
Date	Novembre 2024

Aperçu

Cette documentation couvre l'intégralité du processus d'intégration du paiement Gateway de D-Money, y compris tous les points de terminaison API, les paramètres, la gestion des webhooks et l'interrogation du statut des commandes.

Introduction

Le portefeuille D-Money est un portefeuille numérique innovant et sécurisé pouvant être utilisé à de multiples fins. C'est comme de l'argent numérique que l'on peut utiliser pour effectuer des transferts d'argent ou traiter tout type de transaction.

Les clients peuvent ajouter de l'argent à leur portefeuille D-Money et effectuer des transactions simples, rapides et sécurisées.

Étapes du Flux de Paiement

Étape 1: Générer le Token d'Authentification

Objectif: Générer un token requis pour les appels API ultérieurs.

Options d'Endpoints (Essayer dans l'ordre jusqu'à ce qu'un fonctionne)

POST /apiaccess/payment/gateway/payment/v1/token

POST /payment/gateway/payment/v1/token

En-têtes de Requête

```
{
  "X-APP-Key": "452fe2b7-4105-4fc8-b002-937d10a970b1",
  "Content-Type": "application/json"
}
```

Corps de Requête

```
{
  "appSecret": "5558342c7ac592d627032db14ed51e58"
}
```

Réponse (Succès)


```
{
  "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
  "expires_in": 3600,
  "code": "0",
  "message": "Success"
}
```


Réponse (Erreur)


```
{
  "code": "1001",
  "message": "Invalid credentials",
  "error": "Authentication failed"
}
```

Étape 2: Créer une Commande de Paiement

Objectif: Créer une commande de paiement et obtenir le prepay_id pour la génération de l'URL de checkout.

Options d'Endpoints (Essayer dans l'ordre jusqu'à ce qu'un fonctionne)

POST /apiaccess/payment/gateway/payment/v1/merchant/preOrder

POST /payment/gateway/payment/v1/merchant/preOrder

En-têtes de Requête

{

"Content-Type": "application/json",

"Authorization": "Bearer {token_de_l_etape_1}",

"X-APP-Key": "452fe2b7-4105-4fc8-b002-937d10a970b1"

}

Paramètres de Requête

Parameters	Type	Maximum length	Mandatory	Description
body				
timestamp	String	13	Mandatory	The matching rule is ^[0-9]*[1-9][0-9]*$. This parameter indicates the request sending time and UTC timestamp, in seconds.
method	String		Mandatory.	Method. The value is payment.preorder.
nonce_str	String	32	Mandatory.	The matching rule is \S+. The value is a random string of a maximum of 32 characters, including uppercase letters, lowercase letters, and digits, but excluding special characters.
sign_type	String		Mandatory.	Signature type. The value is SHA256WithRSA.
sign	String	512	Mandatory.	The matching rule is \S+ and the signature is signed.
version	String	4	Mandatory.	The matching rule is \S+, and the interface version number is 1.0.
biz_content	Object			
notify_url	String	512	Mandatory.	Callback address for receiving payment notifications after the payment is successful. The matching rule is \S+.

appid	String	32	Mandatory.	Matching rule: ^[A-Za-z0-9]+$, which indicates the merchant ID allocated to the merchant after registration.
merch_code	String	16	Mandatory.	Matching rule: ^[A-Za-z0-9]+$, indicating the merchant short code allocated to the merchant after registration.
merch_order_id	String	64	Mandatory.	Matching rule: ^[A-Za-z0-9]+$, which indicates the order number generated by the merchant The value must be a string of letters, digits, and underscores (_). Special
trade_type	String		Mandatory.	Enumerated value related to the service scenario. The value is Checkout in the web page payment scenario.
title	String	512	Mandatory.	Matching rule: [^~`!#$%^*()\\-+=|/<>?;:\'"\\[\\]{}\\\\&]*, offering name.
total_amount	String	20	Mandatory.	The matching rule is ^((0{1}\.\d{1, 2})|([1-9]\d*\.{1}\d{1, 2})|([1-9]+\d*))$, which indicates the total order amount. A maximum of two decimal places are allowed.
trans_currency	String	3	Mandatory.	The matching rule is \S+, which complies with the international standard three-letter code, for example: FDJ, USD…etc.
timeout_express	String	10	Mandatory.	The matching rule is ^[0-9]*m$, which indicates the allowed last payment time of an order. The deal will close after the cut-off date. The value ranges from 1 to 120 minutes. The parameter value cannot contain a decimal point. For example, 1.5 hours need to be converted into 90 minutes. The default value is 120 minutes.
business_type	String	32	Optional.	Matching rule: \S+. Enumerated value related to the business scenario. The default value is BuyGoods, indicating the scenario where an individual pays to a merchant directly.
callback_info	String		Optional.	Additional information during callback when the matching rule is \S+.
wallet_reference_data **NOTE** Wallet reference data, for example, {"name": "tom"}.				
key	String		Optional	Displays additional information in key-value format.
value	String		Optional	Value corresponding to the key value.

Exemple de Corps de Requête


```
{
  "nonce_str": "abcd1234efgh5678ijkl9012mnop3456",
  "biz_content": {
    "trans_currency": "DJF",
    "total_amount": "1000",
    "merch_order_id": "1685123456789",
    "appid": "1293049431398401",
    "merch_code": "9988",
    "timeout_express": "120m",
    "trade_type": "Checkout",
    "notify_url": "https://shop.d-moneyservice.dj/webhooks/payment",
    "redirect_url": "https://shop.d-moneyservice.dj/payment/redirect/123",
    "title": "Paiement de Commande",
    "business_type": "OnlineMerchant",
    "callback_info": "oid-123_uid-456"
  },
  "method": "payment.preorder",
  "version": "1.0",
  "sign_type": "SHA256WithRSA",
  "timestamp": "1685123456",
  "sign": "signature_encodee_base64"
}
```

Processus de Génération de Signature

1. **Paramètres pour la signature** (exclure sign, sign_type, biz_content):

appid=1293049431398401&business_type=OnlineMerchant&merch_code=9988&merch_order_id=1685123456789&method=payment.preorder&nonce_str=abcd1234efgh5678ijkl9012mnop3456&notify_url=https://shop.d-moneyservice.dj/webhooks/payment&redirect_url=https://shop.d-moneyservice.dj/payment/redirect/123&timeout_express=120m&timestamp=1685123456&title=Paiement de Commande&total_amount=1000&trade_type=Checkout&trans_currency=DJF&version=1.0

2. **Signer en utilisant RSA-PSS avec SHA256 et MGF1**
3. **Encoder la signature en Base64**

Réponse (Succès)


```
{
  "code": "0",
  "message": "Success",
  "biz_content": {
    "prepay_id": "wx123456789abcdef",
    "order_id": "dmoney_order_12345",
    "status": "created"
  },
  "timestamp": "1685123456",
  "nonce_str": "xyz789"
}
```

Réponse (Erreur)

```
{
  "code": "1002",
  "message": "Invalid order data",
  "error_details": "Missing required field: total_amount"
}
```

Étape 3: Générer l'URL de Checkout

Objectif: Créer l'URL de paiement finale pour rediriger les clients.

Format de l'URL

https://pg.d-moneyservice.dj:38443/payment/web/paygate?{parametres}&sign={signature}&sign_type=SHA256WithRSA&version=1.0&trade_type=Checkout&language=en

Paramètres pour l'URL

Parameter	Description	Type	Required
appid	Unique Merchant ID assigned by the payment gateway system.	string	Yes
merch_code	Merchant code assigned by the payment gateway.	string	Yes
nonce_str	A unique random string that includes uppercase letters, lowercase letters, and numbers.	String	Yes
prepay_id	Prepayment ID obtained from the pre-order API response.	string	Yes

timestamp	The UTC timestamp (in seconds) when the request is generated.	string	Yes
sign	The RSA signature generated for the request.	string	Yes
sign_type	Signature type. Supported value: `SHA256WithRSA`.	string	Yes
version	API version. Current supported value: 1.0.	string	Yes
trade_type	Type of transaction. Supported value: Checkout for web payment.	string	Yes
language	Language of the payment interface. Supported values: en, fr, etc.	string	Yes

Exemple d'URL de Checkout

https://pg.d-moneyservice.dj:38443/payment/web/paygate?appid=1293049431398401&merch_code=9988&nonce_str=abcd1234&prepay_id=wx123456789abcdef&timestamp=1685123456&sign=signature_encodee_url&sign_type=SHA256WithRSA&version=1.0&trade_type=Checkout&language=en

Étape 4: Requête de Statut de Commande

Objectif: Interroger le statut actuel et les informations d'une commande de paiement.

Interface de Requête de Commandes

Description de l'Interface Fonction: Interroger des informations telles que le statut de la commande. **Mot-clé HTTP**: POST **URL de Requête**: https://host:38443/apiaccess/payment/v1/merchant/queryOrder

Endpoint

POST /apiaccess/payment/v1/merchant/queryOrder

En-têtes de Requête

```
{
  "X-APP-Key": "452fe2b7-4105-4fc8-b002-937d10a970b1",
  "Authorization": "Bearer {token_de_l_etape_1}",
  "Content-Type": "application/json"
}
```

Paramètres de Requête

Parameter	Description	Data Type	M/O
Header			
X-APP-Key	App ID registered by a third-party system in the API Fabric.	String	O
Authorization	Authentication application key used by a third-party application to invoke an interface.	String	O
Body			
timestamp	Time when a request is sent. The value is a UTC timestamp, in seconds.	String	M
method	Method. The value is **payment.queryorder**.	String	M

Parameter		Description	Data Type	M/O
nonce_str		The value is a random character string that contains uppercase letters, lowercase letters, and digits but does not contain special characters.	String	M
sign_type		Signature type. The value is SHA256WithRSA.	String	M
sign		Signature.	String	M
version		Interface version number. The value is 1.0.	String	M
biz_content				
	appid	Merchant ID allocated to a merchant after the merchant is registered.	String	M
	merch_code	Short code allocated to a merchant after the merchant is registered.	String	M
	merch_order_id	Order ID generated by a merchant. The value must be letters, digits, and underscores (_). Other special characters are not allowed.	String	M

Exemple de Corps de Requête


```
{
  "timestamp": "1685123456",
  "method": "payment.queryorder",
  "nonce_str": "abcd1234efgh5678ijkl9012mnop3456",
  "sign_type": "SHA256WithRSA",
  "sign": "signature_encodee_base64",
  "version": "1.0",
  "biz_content": {
    "appid": "1293049431398401",
    "merch_code": "9988",
    "merch_order_id": "1685123456789"
  }
}
```

Génération de Signature pour Requête

1. **Paramètres pour la signature** (exclure sign, sign_type, biz_content):

appid=1293049431398401&merch_code=9988&merch_order_id=1685123456789&method=payment.queryorder&nonce_str=abcd1234efgh5678ijkl9012mnop3456&timestamp=1685123456&version=1.0

2. **Signer en utilisant RSA-PSS avec SHA256 et MGF1**
3. **Encoder la signature en Base64**

Réponse (Succès)


```
{
 "result": "SUCCESS",
 "code": "0",
 "msg": "Query successful",
 "sign": "signature_reponse",
 "nonce_str": "xyz789",
 "sign_type": "SHA256WithRSA",
 "biz_content": {
  "merch_order_id": "1685123456789",
  "order_status": "PAID",
  "payment_order_id": "dmoney_order_12345",
  "trans_time": "2024-05-27 14:30:25",
  "trans_currency": "DJF",
  "total_amount": "1000"
 }
```

}

Réponse (Erreur)

```
{
  "result": "FAIL",
  "code": "1004",
  "msg": "Order not found",
  "sign": "signature_reponse",
  "nonce_str": "xyz789",
  "sign_type": "SHA256WithRSA"
}
```

Étape 5: Gestion du Webhook de Paiement

Objectif: Recevoir et traiter les notifications de statut de paiement de D-Money.

Endpoint du Webhook

POST /webhooks/payment

Paramètres de Requête Webhook

Parameter	Description	Data Type	M/O
Body			
notify_url	Callback address for receiving payment notifications after successful payment.	String	O
appid	Merchant ID assigned to a merchant after the merchant is registered.	String	O
notify_time	Notification sending time, in seconds. The time format is UTC.	String	O
merch_code	Short code assigned to a merchant after the merchant is registered.	String	O

Parameter	Description	Data Type	M/O
merch_order_id	Order ID generated by a merchant.	String	O
payment_order_id	Order ID generated by the platform.	String	O
total_amount	Order payment amount in a transaction.	String	O
trans_currency	Transaction currency type.	String	O
trade_status	Transaction status. The options are as follows: • **Paying**: The user has paid, but the transaction such as coupons has been agreed to pay. • **Expired**: The payment is overdue and unpaid. • **Completed**: The payment is complete. • **Failure**: The payment fails.	String	O
trans_end_time	Transaction end time, in seconds.	String	O
callback_info	Callback information. If this parameter is transferred when a merchant places an order, a notification is asynchronously returned to the merchant.	String	O
sign	Signature information.	String	O
sign_type	Signature type. The value is SHA256WithRSA.	String	O

Exemple de Requête Webhook


```
{
  "merch_order_id": "1685123456789",
  "trade_status": "completed",
  "transId": "dmoney_trans_12345",
  "total_amount": "1000",
  "trans_currency": "DJF",
  "callback_info": "oid-123_uid-456",
  "timestamp": "1685123800",
  "sign": "signature_de_dmoney",
  "sign_type": "SHA256WithRSA"}
```


Réponse Webhook (Succès)

```
{
  "success": true,
  "message": "Notification de paiement traitée avec succès"
}
```

Réponse Webhook (Erreur)

```
{
  "error": "Commande non trouvée",
  "message": "Aucune commande trouvée avec merchant_order_id: 1685123456789"
}
```

Schéma de Base de Données

Migration de la Table Orders

```
Schema::create('orders', function (Blueprint $table) {
    $table->id();
    $table->foreignId('user_id')->constrained()->cascadeOnDelete();
    $table->string('order_number')->unique();
    $table->string('merchant_order_id')->unique();
    $table->decimal('total', 10, 2);
    $table->enum('payment_status', ['pending', 'completed', 'failed'])->default('pending');
    $table->text('payment_token')->nullable();
```

```
    $table->string('payment_transaction_id')->nullable();
    $table->string('transaction_id')->nullable();
    $table->json('payment_response')->nullable();
    $table->text('checkout_url')->nullable();
    $table->text('redirect_url')->nullable();
    $table->foreignId('status_id')->default(1);
    $table->timestamps();
});
```

Migration de la Table Order Logs

```
Schema::create('order_logs', function (Blueprint $table) {
    $table->id();
    $table->foreignId('order_id')->constrained()->cascadeOnDelete();
    $table->text('message');
    $table->json('data')->nullable();
    $table->string('type')->default('general');
    $table->timestamps();
});
```

Considérations de Sécurité

1. Vérification de Signature

Toujours vérifier les signatures de webhook:

```
public function verifyCallbackSignature($data, $signature)
{
    $callbackParams = [];
    $excludedParams = ['sign', 'sign_type'];

    foreach ($data as $key => $value) {
        if (!in_array($key, $excludedParams) && $value !== null && $value !== '') {
            $callbackParams[$key] = $value;
        }
    }

    $stringToVerify = $this->createRawRequest($callbackParams);
    // Vérifier en utilisant la clé publique de D-Money
    return $this->verifySignature($stringToVerify, $signature);
}
```

2. Sécurité d'Environnement

- Stocker les clés privées de manière sécurisée
- Utiliser des variables d'environnement pour les données sensibles
- Implémenter la limitation de taux sur les endpoints de webhook
- Utiliser HTTPS pour toutes les communications

3. Idempotence

Gérer les notifications de webhook dupliquées:

public function handlePaymentNotification(Request $request)

```
{
    $data = $request->all();
    $merchantOrderId = $data['merch_order_id'];

    // Vérifier si déjà traité
    $existingLog = OrderLog::where('order_id', $orderId)
        ->where('type', 'payment_notification')
        ->where('data->transId', $data['transId'])
        ->first();

    if ($existingLog) {
        return response()->json(['message' => 'Déjà traité']);
    }

    // Traiter la notification...
}
```

Tests

Environnement de Test

DMONEY_BASE_URL=https://pgtest.d-money.dj:38443

Utiliser les identifiants de test fournis par D-Money

Étapes de Flux de Test

1. **Générer Token**: Tester avec des identifiants valides/invalides
2. **Créer Commande**: Tester avec divers montants et devises de commande
3. **Interroger Statut**: Tester la vérification du statut de commande
4. **Simuler Webhooks**: Tester l'endpoint webhook avec des payloads d'exemple
5. **Gérer Redirections**: Tester les pages de redirection succès/échec

Données de Test Exemple

```
// Données de commande de test
$testOrderData = [
    'trans_currency' => 'DJF',
    'total_amount' => '100',
    'merch_order_id' => '1685123456789',
    'title' => 'Commande de Test',
    'callback_info' => 'oid-1_uid-1'
];

// Payload webhook de test
$testWebhookData = [
    'merch_order_id' => '1685123456789',
    'trade_status' => 'completed',
    'transId' => 'test_trans_123',
    'total_amount' => '100',
    'trans_currency' => 'DJF'
```

];

Dépannage

Problèmes Courants et Solutions

1. Échec de Génération de Token

Symptômes: Erreurs 401/403 sur l'endpoint de token **Solutions**:

- Vérifier APP_KEY et APP_SECRET
- Vérifier différents chemins d'endpoint
- S'assurer de l'en-tête Content-Type approprié

2. Échec de Création de Commande

Symptômes: Erreurs de signature invalide ou paramètre manquant **Solutions**:

- Valider tous les champs biz_content requis
- Vérifier le processus de génération de signature
- Vérifier l'ordre des paramètres dans la chaîne de signature

3. Webhook Non Reçu

Symptômes: Statut de commande non mis à jour après paiement **Solutions**:

- Vérifier l'accessibilité de l'URL webhook depuis les réseaux externes
- Vérifier la validité du certificat SSL
- Vérifier les logs du serveur pour les requêtes entrantes
- Tester l'endpoint webhook manuellement

Conclusion

Cette documentation fournit un guide complet pour l'intégration de la Payment Gateway de D-Money. Suivez les étapes dans l'ordre, mettez en œuvre une gestion des erreurs appropriée et assurez-vous que les bonnes pratiques de sécurité sont respectées tout au long de l'intégration.

Pour obtenir une assistance supplémentaire ou des questions sur des détails d'implémentation spécifiques, consultez la documentation officielle de l'API D-Money ou contactez l'équipe d'assistance.