

Les API de paiement

Table des matières

I. Paiement en ligne	3
II. Exercice : Quiz	7
III. Utilisation d'une API de paiement	8
IV. Exercice : Quiz	15
V. Essentiel	16
VI. Auto-évaluation	16
A. Exercice	16
B. Test	17
Solutions des exercices	17

I. Paiement en ligne

Durée : 1 h

Environnement de travail : un pc

Contexte

Les méthodes de consommation des personnes n'ont cessé d'évoluer avec le temps et il est devenu commun de réaliser des achats sur internet.

De l'achat de service par abonnement (comme la vidéo, la musique à la demande) ou même des services ponctuels (comme de la conception web, ou sur d'autre domaine d'activité), jusqu'à l'achat de produits physiques, les ventes passent toujours plus par internet.

Cependant, bien que la plateforme web soit très attractive, car elle ne limite pas géographiquement les ventes, elle possède le gros défaut de compliquer les paiements, c'est ainsi que de nombreuses sociétés se sont spécialisées dans cette activité.

Aujourd'hui il est possible de créer des modules de paiements par un service indépendant ou par la banque du vendeur, dans la majorité des cas ces modules passent par une API.

Dans ce cours nous verrons, le concept de paiement par internet et verront l'utilisation des APIs associés à travers 2 exemples, grâce au service de Stripe et PayPal.

Le commerce en ligne étant devenu partie intégrante des consommateurs, il a été nécessaire de mettre en place des solutions de paiement pour ces derniers.

Seulement, ce système n'est pas aussi simple car les transactions sont vérifiées par les banques et font déjà l'objet d'un protocole très spécifique.

Il existe heureusement des solutions préconçues par les banques pour les paiements par carte bleue, mais aussi par d'autres sociétés servant soit d'intermédiaire, soit de banque spécialisée.

Le paiement se déroule alors ainsi pour l'utilisateur : API de la banque :

- Le client arrive sur une page d'information bancaire (ou il rentre sa carte bleue) - Cette page appartient soit à l'application soit à la société de paiement.
- Le client doit s'authentifier auprès de sa banque (par SMS ou application)
- Le client retourne sur le site avec le paiement valide ou non

Remarque

En France, l'authentification auprès de la banque (pour les montants supérieurs à 30€) est devenue pour les nouveaux paiement sur internet obligatoire depuis le 15 mai 2021, il s'agit de l'authentification forte comme visible sur cette page : RÉPUBLIQUE FRANÇAISE¹

¹ <https://www.service-public.fr/particuliers/actualites/A14910>

Le client se retrouve donc avec un seul procédé, peu importe l'API exploitée.

Plus récemment, c'est le prélèvement sur compte bancaire qui a fait son apparition et pourrait prochainement devenir une norme (Solution proposée par Amazon, avec les Carte bleu par exemple).

En tant que développeur, il est donc primordial de bien choisir la solution qui sera exploitée. À cela se pose 3 cas de figure :

- Le développeur ou le client demandant la prestation possède une banque proposant une API et veut éviter les frais supplémentaires,
- Le développeur utilise une solution par prédilection,
- La plate-forme du site supporte uniquement une solution.

Dans ces cas-là, 1 seule solution sera possible, et dépendra, du développeur, du client ou de la plate-forme.

Parmi les acteurs proposant des solutions de paiements en ligne, sont notable :

- PayPal, un mastodonte du paiement en ligne, utilisé sur de nombreux site.
- Stripe, solution plus récente, ayant énormément communiqué et étant exploiter par des entreprises comme Deliveroo ou Adidas.
- Mango Pay, une solution sécurisée, particulièrement intéressante pour le séquestre de montant (Marketplace par exemple) et pour les paiements. Cette dernière est utilisée par Vinted et Rue du commerce.

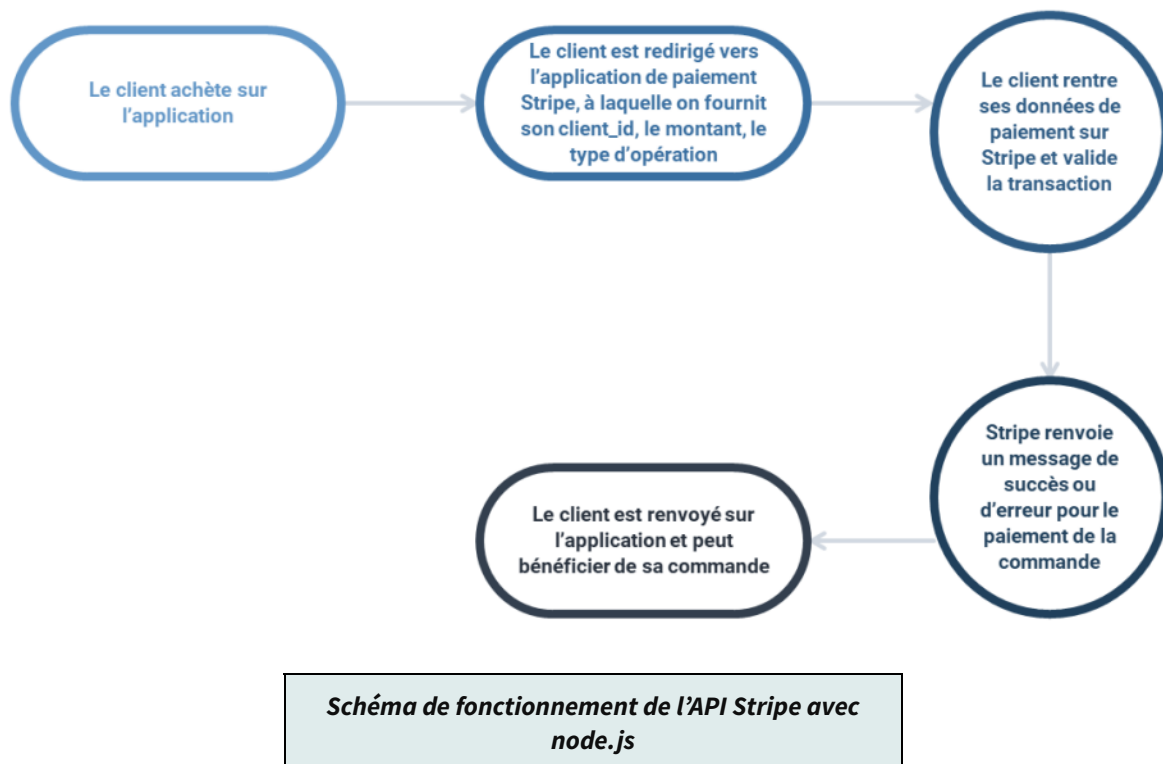
Remarque

Les 3 plates-formes citées ne sont qu'un échantillon, mais il est à noter que les 3 supportent les principales carte bleu (Visa, Mastercard et CB) et proposent un système de paiement récurrent.

En pratique, le fonctionnement dépend de la plateforme mais respecte grossièrement un schéma :

- Le commerçant ou développeur créé un compte chez le fournisseur de service,
- Il obtient une clé privée (parfois un id client et une clé),
- Le client choisit l'option du service pour payer,
- Le client est redirigé vers le service (la page transmet alors : un identifiant de commande, le montant, et la clé),
- Le client paye chez le fournisseur de service,
- Le service renvoi une requête acceptée ou rejetée à l'application avec un identifiant unique de transaction,
- Le client est renvoyé vers l'application.

Pour prendre un cas plus concret, un paiement chez Stripe avec node.js et React se déroule de la manière suivante :



Bien sûr, il est aussi possible de réceptionner les données de paiements du client directement sur l'application, et les communiquer aux services, mais ce fonctionnement s'est compliqué depuis la loi sur l'authentification forte, il est donc primordial dans le cas d'une réception des informations par l'application, de prévoir, l'authentification forte.

Le paiement se fait alors avec une étape d'authentification supplémentaire (c'est l'équivalent de l'authentification à 2 facteurs), ainsi après avoir envoyé les données de sa carte bleue, le client doit s'identifier via un code unique qu'il obtient par :

- Sms
- Mail
- Application
- Générateur

À ce niveau-là et avant l'obtention de ce code unique, le paiement n'est pas validé.

Attention

Le choix du prestataire qui s'occupera du paiement, de la méthodologie et des moyens de paiement acceptés doit faire l'objet d'une grande attention. Selon les termes des prestataires par exemple, il est possible de perdre tous les droits d'une API à la suite d'un produit inadapté (prendre exemple : vendre un bon d'achat en ligne avec SumUp, autre plateforme de paiement par TPE et en ligne).

Exemple

La mise en place d'un paiement avec Stripe via PHP.

Pour commencer il est nécessaire d'installer les composants de Stripe via « *Composer* » grâce à la ligne de commande :

```
1 $ composer require stripe/stripe-php
```

Il est ensuite nécessaire d'importer les fonctions et de déclarer la clé API enregistrée sur le site avec la ligne :

```
1 require 'vendor/autoload.php'; \Stripe\Stripe::setApiKey('sk_test_VePHdqKTYQjKNInc7u56JBrQ');
```

Ici la clé API est donc « `sk_test_VePHdqKTYQjKNInc7u56JBrQ` ». Il faut ensuite définir, créer la session de paiement du panier, avec par exemple les lignes suivantes :

```
1 $checkout_session = \Stripe\Checkout\Session::create([
2   'line_items' => [[
3     'price_data' => [
4       'currency' => 'EUR',
5       'product_data' => [
6         'name' => 'my_product',
7       ],
8       'unit_amount' => 2000,
9     ],
10    'quantity' => 1,
11  ]],
12  'mode' => 'payment',
13  'success_url' => 'http://localhost/success.html',
14  'cancel_url' => 'http://localhost/cancel.html',
15 ]]);
```

Ici est défini, la création d'un checkout (passage à la caisse) et dedans se trouve :

Ici est défini, la création d'un checkout (passage à la caisse) et dedans se trouve :

- Line_items : qui constitue les informations des produits vendus,
- Mode : qui est la méthode appelé (ici paiement, pour un paiement ponctuel),
- Success_url : qui définit l'adresse à appeler en cas de paiement validé,
- Cancel_url : qui définit l'adresse à appeler en cas de paiement non valide.

Ici line_items contient un tableau où chaque ligne correspond à un produit et une quantité.

Le produit est défini par la clé price_data qui contient les valeurs :

- Currency : définissant le type de valeur utilisé (ici les euros),
- Product_data : qui contiendra les informations du produit comme le nom,
- Unit_amount : qui contiendra le prix à l'unité en cents (ici 20,00 €).

Capture d'écran interface de paiement Stripe

La page de paiement s'affiche alors en effectuant une requête POST vers cette adresse.

Remarque

Dans l'exemple ci-dessus, le mode de paiement est « *payment* » qui consiste en un paiement unique.

2 autres modes existent avec Stripe :

- *subscription* : qui permet de créer un paiement récurrent (par exemple, pour un abonnement).
- *setup* : qui permet de créer une configuration pour un futur paiement client (par exemple, un paiement en 2 temps par un client, *setup* permet de configurer le second paiement).

Les nuances d'utilisations de l'API entre chaque langage peuvent être visible sur la documentation à cette adresse : Démarrage rapide¹

Par exemple, pour le même exemple avec node.js, chaque item sera stocké dans un objet et la clé API pourra être spécifiée lors de l'import du module Stripe.

De même, il existe des composants pour le front en HTML, React et Next.js, ainsi que pour le back en .NET, Go, Ruby, Node.js, PHP, Java et Python.

Les solutions sont donc proposées pour de multiples configuration et parfois, sous plusieurs versions.

Remarque

Pour la configuration PHP vue précédemment, il est possible d'installer les modules de Stripe sans composer, dans ce cas, il est nécessaire de récupérer le code sur leur GitHub.

Rappel

Comme pour chaque API, la documentation officielle de cette dernière reste l'élément le plus important.

Dès lors, que le développeur a la capacité d'analyser, comprendre et utiliser cette dernière, il devient aisé d'utiliser l'API associée.

Exercice : Quiz

[solution n°1 p.19]

Question 1

Le service de paiement dépend forcément de la banque ou la société de la personne réceptionnant le paiement

- ☐ Faux
- ☐ Vrai

Question 2

En France, les paiements en lignes nécessitent une authentification forte si le montant est supérieur à :

- ☐ 10 €
- ☐ 40 €
- ☐ 20 €
- ☐ 30 €

Question 3

¹ <https://stripe.com/docs/checkout/quickstart>

Dans beaucoup de cas (comme Paypal ou Stripe) le client indique ces informations bancaires :

- ☐ Sur le site du prestataire de paiement
- ☐ Sur l'application

Question 4

L'utilisation de Stripe sur PHP de base nécessite :

- ☐ Composer
- ☐ Symfony
- ☐ cURL
- ☐ Pas de module complémentaire

Question 5

La première ligne à fournir à l'API Stripe est

- ☐ La clé API
- ☐ Le panier et sa création

III. Utilisation d'une API de paiement

Dans le chapitre précédent a été présenté rapidement Stripe, cette solution est parfaite pour un usage en tant qu'auto entrepreneur par exemple ou pour des commerces locaux.

Lorsque les ventes du client nécessitent une garantie pour les produits ou un système de paiement plus avancé, il peut être intéressant de passer par une solution comme PayPal, ce chapitre présente comment utiliser l'API de PayPal pour effectuer des paiements standard.

Remarque

D'autres solutions proposent d'assurer les produits et le client et sont reconnus, mais PayPal reste une valeur particulièrement connue du grand public et reconnue par ce dernier.

PayPal, comme tous les services de paiements, nécessite une inscription en tant que vendeur ou développeur sur leur site. Pour cela il est nécessaire de se rendre sur la page de leur service : PayPal¹

Vous pouvez vous inscrire en sélectionnant « *ouvrir un compte professionnel* » et en suivant les diverses étapes (courriel, mot de passe, nom, prénom, dénomination sociale, etc.).



À la fin du parcours, PayPal indique les premières étapes pour l'utilisation de leur API :

Inviter votre développeur



Ajoutez votre développeur à votre compte et envoyez un email contenant des instructions sur la procédure à suivre.

[Inviter votre développeur](#)



Votre développeur pourra



Configurer PayPal Checkout avec les paiements par carte bancaire de base
Utilisez les documents pour les développeurs pour créer votre code PayPal Checkout. Ajoutez ce code à votre site ou application pour permettre à vos clients de choisir leur mode de paiement. [En savoir plus](#)



Configurer la facturation à l'aide d'API
Envoyez une facture ou un devis en quelques minutes. Vos clients peuvent payer par carte ou avec PayPal. [En savoir plus](#)



Configurer des abonnements à l'aide d'API
Proposez des abonnements sur votre site pour avoir des revenus plus stables et plus prévisibles. [En savoir plus](#)

Capture d'écran de la fin de procédure d'enregistrement sur PayPal

Il est donc nécessaire ensuite de récupérer les composants nécessaires à la mise en place d'un passage caisse sur l'application (checkout).

Pour cela, la première étape de configuration des composants de l'API sera de récupérer les identifiants d'environnement de test.

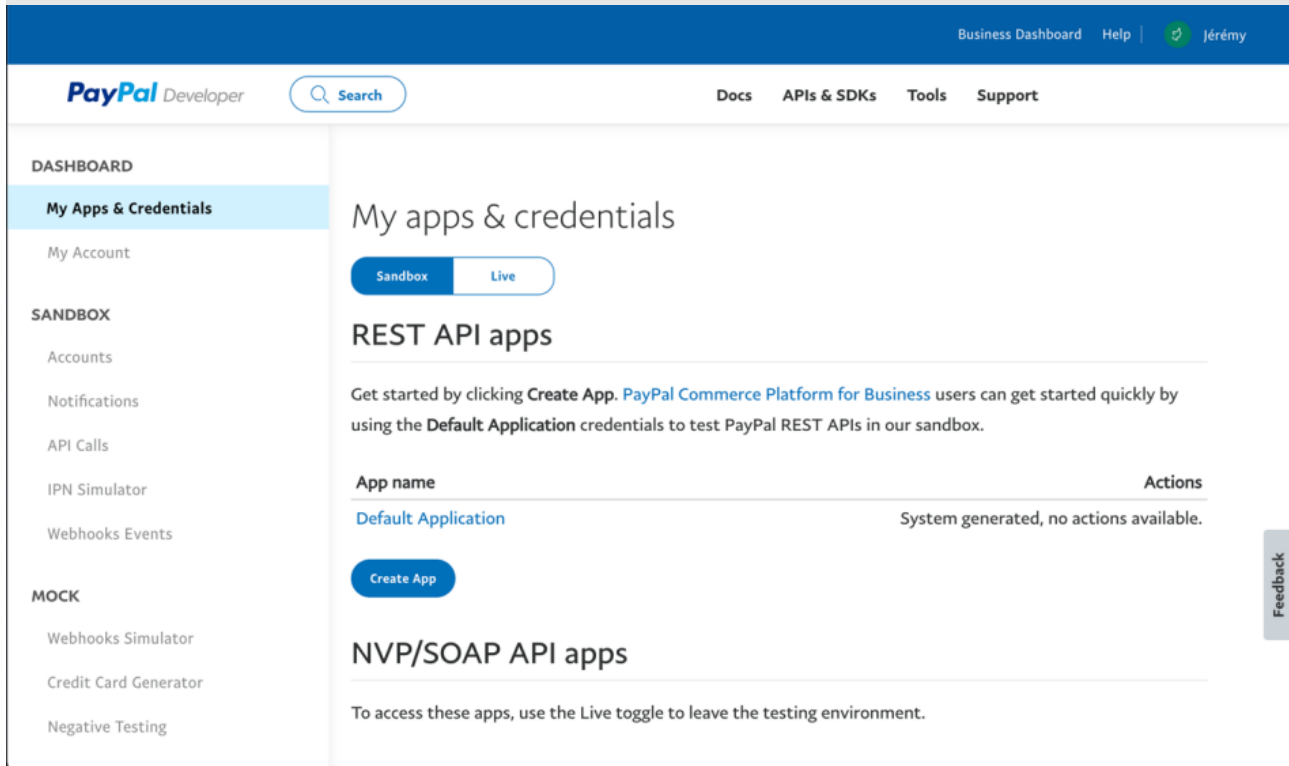
Ces identifiants permettent de tester la configuration sans créer pour autant de vraie transaction.

¹ <https://www.paypal.com/fr/webapps/mpp/account-selection>

Méthode

La création des identifiants de test se fait via l'espace développeur disponible à cette adresse : My apps & credentials¹

Après la connexion, le tableau de bord développeur s'affiche :



Capture d'écran du dashboard developer PayPal

La partie intéressante ici est de voir sous « My apps & credentials » que l'interface est en mode Sandbox (en français « bac à sable », ou « terrain de test »).

Il est possible de créer de nouveaux identifiants à partir de la partie « Accounts » sous SANDBOX dans le menu de gauche, puis en cliquant sur « Create Account ».

¹ <https://developer.paypal.com/developer/applications>

Une modal s'affiche alors demandant les principales informations :

Capture d'écran procédure création de faux compte sur PayPal

Une fois le compte développeur créé, le nouveau compte apparaît dans la liste (sous forme de tableau) et les informations sont disponibles grâce aux 3 points et au bouton « *view / edit account* », pour obtenir la page ci-dessus.

À partir de là, un onglet « *API Credentials* » permet d'obtenir les informations qui seront indispensables pour développer notre liaison avec l'API.

Sur cet onglet seront visibles les informations pour le compte test :

- L'adresse électronique
- Le mot de passe
- La signature

Exemple

Pour utiliser l'API avec par exemple, node.js il sera nécessaire d'installer le SDK PayPal spécifique à node.

Dans un projet avec node initialisé (à la suite de la commande `npm init`), il sera nécessaire de lancer la commande suivante depuis le dossier du projet :

```
1 $ npm install paypal-rest-sdk
```

Cette commande va installer la dernière version du sdk paypal.

Ensuite dans le fichier js l'utilisant, il sera nécessaire d'importer le package et de notifier la configuration :

```
1 const paypal = require('paypal-rest-sdk')
2 paypal.configure({
3   'mode': 'sandbox',
4   'client_id':
5     'Aa1d1ADmueruF75292ygtbJTHC7PY9Jg_LNza9MXXdFJq6jcGACLYb1W7cWcVnu4Hcrp8lUCHF_Y4Uh',
6   'client_secret': 'ENZNAxIRWKwE_lqhTs9L-o_zA8HwSAc2BH7Wct5tEid7SKhhPeP3XWjhQW9k7Z_ehwjaSyr-PGBU3u4K'
7 })
```

Ici est déclaré le mode, il peut s'agir de :

- Sandbox : mode de test se basant sur la SandBox de PayPal
- Live : mode d'utilisation par les utilisateurs

Sont aussi déclaré le `client_id` et `client_secret`, 2 informations trouvable sur le site développeur, dans le *dashboard* en cliquant sur l'application (de base, la seule application existante est « *Default Application* »).

Attention

Une bonne pratique est de bien créer une nouvelle application pour chaque application exploitant le SDK, chacune doit posséder son `client_id` et son secret.

De même, le secret doit être régénéré dès lors qu'une faille aurait pu être détectée.

Exemple

Il est ensuite nécessaire de créer l'adresse qui appellera les services de PayPal afin d'effectuer le paiement via le code suivant :

```
1 app.post('/checkout-paypal', (req, res) => {
2   const paymentInfo = {
3     "intent": "sale",
4     "payer": {
5       "payment_method": "paypal"
6     },
7     "redirect_urls": {
8       "return_url": "http://localhost:4343/success",
9       "cancel_url": "http://localhost:4343/cancel"
10    },
11    "transactions": [{
12      "amount": {
13        "currency": "EUR",
14        "total": "20.00"
15      }
16    }]
17  }
```

Ici, une requête à l'adresse `adresse/checkout-paypal` avec la méthode POST, créera un objet compatible avec les attentes de l'API PayPal avec les champs suivants :

- Intent : l'intention de l'objet (ici une vente grâce au mot clé `sale`)
- Payer : la méthode de paiement
- Redirect_urls : les urls de réussites ou d'échec
- Transactions : les informations de transaction

Dans cet exemple, Transaction ne contient que Amount, qui consiste en le total de la transaction mais il est possible de détailler les produits de la manière suivante :

```
1 "item_list": {
2   "items": [{
3     "name": "my_product",
4     "sku": "001",
5     "price": "20.00",
6     "currency": "EUR",
7     "quantity": 1
8   }]
9 },
```

Ici, `item_list` est un objet contenant items qui listera les produits au format objet dans un array.

Il est alors possible de détailler les produits avec le nom (`name`), son identifiant (`sku`), son prix (`price`), la devise (`currency`) et la quantité (`quantity`).

La liste des items (`item_list`) se situe dans la partie transaction avant l'objet « *amount* ».

Ensuite, il est nécessaire de créer le paiement avec le SDK PayPal grâce à l'objet créé précédemment de la manière suivante :

```
1 paypal.payment.create(paymentInfo, function (error, payment) {
2     if (error) {
3         throw error;
4     } else {
5         for(let i = 0; i < payment.links.length; i++){
6             if(payment.links[i].rel === 'approval_url'){
7                 res.redirect(payment.links[i].href);
8             }
9         }
10    }
11 })
```

Ici, le code crée un paiement sur le SDK PayPal qui reçoit en paramètre l'objet créé précédemment, il attend alors une erreur ou un objet de paiement qui contiendra une liste de liens de redirection dont celui permettant le paiement (`approval_url`).

L'utilisateur arrive alors sur la page de paiement suite à la redirection effectuée par le serveur avec la ligne : `res.redirect`

Une fois l'utilisateur connecté, il se retrouvera sur la page de validation comme ci-dessous et sera renvoyé sur la page de succès ou d'échec selon le résultat du paiement.

Test Store



20,00 EUR

Bonjour John [Déconnexion](#)

Adresse de livraison

John Doe
Av. de la Pelouse, 75002 Paris

[Modifier](#)

Payer avec



Solde PayPal

20,00 EUR

☐

Définir comme source d'approvisionnement préférée



Visa
Crédit ••••8177



Rabobank Nederland
Courant ••••1955

Capture d'écran de paiement par faux compte sur PayPal

À la suite de cela, si le paiement est un succès, PayPal envoie 2 informations :

- Le paymentId : l'identifiant de paiement
- Le payerId : l'identifiant du client

Avec ces informations, il est possible de vérifier la transaction avec la fonction du sdk execute depuis le module payment.

Remarque

En mode SandBox, les clients fictifs doivent être créés de la même façon que les comptes professionnels fictif, il est alors possible de récupérer l'identifiant (mail) et le mot de passe du faux client.

Conseil

Bien que l'utilisation du mode live de l'API PayPal soit similaire en tout point, et permettra de mettre en ligne directement un système de paiement fonctionnel, il est recommandé de passer par le mode SandBox pour prévenir d'éventuelles erreurs et vérifier le fonctionnement optimal.

Le mode SandBox permet par ailleurs l'utilisation du paiement sur une version ou une branche développeur.

Ainsi, il est possible grâce à cette méthode de réaliser des paiements via PayPal, énormément d'autres API existent et fonctionnent sur un système similaire.

Dans énormément de cas, un bac à sable ou SandBox est disponible et il est recommandé de l'utiliser avant de développer sur la version de production.

En résumé, les points primordiaux pour la maîtrise d'une API de paiement sont :

- La documentation
- L'environnement bac à sable
- La maîtrise du langage utilisé

La plupart des APIs de paiements sont basées sur un système REST avec un système de jeton propre à l'application, au vendeur ou aux 2.

Le fonctionnement d'une API de paiement**Exercice : Quiz**

[solution n°2 p.20]

Question 1

PayPal a l'avantage :

- ☐ D'assurer les ventes de produits pour les clients
- ☐ D'assurer les ventes de services pour les clients

Question 2

La première étape pour l'utilisation de l'API de paiement est :

- ☐ La mise en place d'un code de test
- ☐ L'inscription sur le site du prestataire

Question 3

PayPal nécessite :

- ☐ 3 informations pour connecter l'application
- ☐ 1 information pour connecter l'application
- ☐ 2 informations pour connecter l'application
- ☐ 4 informations pour connecter l'application

Question 4

La SandBox PayPal permet de créer :

- ☐ De faux compte vendeur
- ☐ De faux compte client
- ☐ De fausses applications

Question 5

Le mode Live sur l'API permet

- ☐ De passer l'API sur des transactions réelles
- ☐ D'obtenir les statistiques en temps réel

V. Essentiel

Fondamental

Bien que grandement facilité par les APIs, la mise en place de paiements en ligne nécessite certaines connaissances, afin de choisir le prestataire idéal.

Cela passe par la plate-forme exploitée par l'application (langage, CMS, Framework, hébergeur) mais aussi par les taux, l'application des lois régionales et les préférences de l'utilisateur.

Une fois le prestataire sélectionné, il est nécessaire de prendre connaissance des modalités d'utilisation qui sont trouvable dans la documentation développeur et de lire cette dernière afin de comprendre la logique d'utilisation de l'API.

Dans le cas de Stripe, par exemple, une unique clé était nécessaire pour s'authentifier, et le détail des achats de base était obligatoire.

Dans le cas de PayPal, 3 clés étaient nécessaires et le détail n'était pas obligatoire.

Il est donc nécessaire de bien comprendre ces différences entre chaque solution, afin de pouvoir l'utiliser de manière optimale.

Enfin, il reste préférable d'utiliser une API en mode test (ou SandBox) dans un premier temps, afin de réaliser des tests sans conséquences derrière.

VI. Auto-évaluation

A. Exercice

Jeune développeur dans une entreprise de vente possédant déjà un compte chez Stripe, le responsable vous donne la clé d'accès et vous demande de mettre en place les composants sur le serveur PHP pour recevoir une demande de paiement pour un produit unique.

Question 1

[solution n°3 p.21]

Codez le back pour effectuer un paiement sur le produit suivant :

Nom : chaussure Geog

Prix : 13,00 €

Avec la clé entreprise suivante :

« PJKI892JDUCNFONZNDOIDZBDIB98_90 »

En vous basant sur l'API, en PHP avec quel outil est-il possible de créer ce paiement ?

Dans une autre entreprise, vous êtes développeur mais n'agissez que sur la création de fonctionnalité, vous utilisez donc leur prestataire de paiement, uniquement en mode Test. Le responsable vous laisse créer de nouveaux identifiants sur leur accès PayPal pour concevoir une fonction.

Question 2

[solution n°4 p.21]

Rendez vous sur le site PayPal developer et créez les nouveaux identifiants puis récupérez les 3 identifiants nécessaires.

Quels sont les identifiants récupérés ?

B. Test**Exercice 1 : Quiz**

[solution n°5 p.23]

Question 1

Les solutions comme PayPal sont optimisées pour les paiements :

- ☐ Unique
- ☐ Par abonnement
- ☐ De test ou SandBox
- ☐ De Marketplace

Question 2

Le développeur peut être contraint sur un API de paiement spécifique par un client.

- ☐ Faux
- ☐ Vrai

Question 3

La partie redirect_urls sur les apis permettent :

- ☐ De définir les adresses de redirection d'échec et de succès.
- ☐ De rediriger la transaction vers un autre lien.

Question 4

Pour les identifiants il est recommandé :

- ☐ D'avoir une unique paire de clé d'application pour toutes les applications.
- ☐ D'avoir une paire de clé d'application spécifique pour chaque application.

Question 5

Les points primordiaux d'utilisation d'une API de paiement sont :

- ☐ Sa documentation
- ☐ Son environnement bac à sable
- ☐ La maîtrise du langage de programmation utilisé
- ☐ Le taux de client déjà existant sur la plateforme

Solutions des exercices

Exercice p. 7 Solution n°1**Question 1**

Le service de paiement dépend forcément de la banque ou la société de la personne réceptionnant le paiement

☒ Faux

Faux, Il peut appartenir à la banque de ce dernier mais peut aussi provenir d'un prestataire.

☐ Vrai

Question 2


En France, les paiements en lignes nécessitent une authentification forte si le montant est supérieur à :

☐ 10 €

☐ 40 €

☐ 20 €

☒ 30 €


 À partir de 30 €, les paiements en ligne nécessitent une authentification forte en France, cela affecte l'utilisation de l'API.

Question 3

Dans beaucoup de cas (comme Paypal ou Stripe) le client indique ces informations bancaires :

☒ Sur le site du prestataire de paiement

☐ Sur l'application

 Dans de nombreux cas, le paiement s'effectue sur le site du prestataire de paiement

Question 4


L'utilisation de Stripe sur PHP de base nécessite :

☒ Composer

☐ Symfony

☐ cURL

☐ Pas de module complémentaire


 L'utilisation basique de Stripe avec PHP nécessite de composer.

Question 5

La première ligne à fournir à l'API Stripe est

☒ La clé API


☐ Le panier et sa création

 La première ligne à fournir à Stripe est la clé API via l'appel de Stripe `::setApiKey`

Exercice p. 15 Solution n°2


Question 1

PayPal a l'avantage :

- ☒ D'assurer les ventes de produits pour les clients
- ☐ D'assurer les ventes de services pour les clients
-  PayPal permet d'assurer les ventes de produits pour les clients.


Question 2

La première étape pour l'utilisation de l'API de paiement est :

- ☐ La mise en place d'un code de test
- ☒ L'inscription sur le site du prestataire
-  La première étape est l'inscription sur le site du prestataire afin de récupérer les clés nécessaires.


Question 3

PayPal nécessite :

- ☒ 3 informations pour connecter l'application
- ☐ 1 information pour connecter l'application
- ☐ 2 informations pour connecter l'application
- ☐ 4 informations pour connecter l'application
-  PayPal nécessite 3 informations pour connecter l'application : le mail, l'id client et l'id secret.

Question 4

La SandBox PayPal permet de créer :

- ☒ De faux compte vendeur
- ☒ De faux compte client
- ☒ De fausses applications
-  La SandBox PayPal permet de simuler les vendeurs, les clients, et les fausses applications.

Question 5

Le mode Live sur l'API permet

- ☒ De passer l'API sur des transactions réelles
- ☐ D'obtenir les statistiques en temps réel

Q Le mode live est l'opposé du mode SandBox, il permet de passer sur des transactions réelles.

p. 16 Solution n°3

Le composant sera à coder de la manière suivante :

```
1 require 'vendor/autoload.php';
2 \Stripe\Stripe::setApiKey('PJKI892JDUCNFONZND0IDZBDIB98_90 ');
3
4 $checkout_session = \Stripe\Checkout\Session::create([
5     'line_items' => [[
6         'price_data' => [
7             'currency' => 'EUR',
8             'product_data' => [
9                 'name' => Chaussure Geog,
10            ],
11            'unit_amount' => 1300,
12        ],
13        'quantity' => 1,
14    ]],
15    'mode' => 'payment',
16    'success_url' => 'http://localhost/success.html',
17    'cancel_url' => 'http://localhost/cancel.html',
18 ])
```

Les 2 outils permettant d'utiliser l'API avec PHP sont : composer et code natif.

p. 17 Solution n°4

Pour cet exercice, il faut se rendre sur le dashboard developper de PayPal à l'adresse suivante :

My apps & credentials¹

Une fois connecté, il est nécessaire de créer une nouvelle application grâce au bouton : « *Create App* » comme ci-dessous



Capture d'écran bouton create app sur le dashboard developper PayPal

Il faut ensuite nommer l'application, puis appuyer sur Create App

¹ <https://developer.paypal.com/developer/applications>

Application Details

App Name

nono

App Type

- ☒ **Merchant** – Accept payments as a merchant (seller)
- ☐ **Platform** – Move payments to sellers as a platform (marketplace, crowdfunding, or e-commerce platform)

Sandbox Business Account

sb-evw0115823720@business.example.c ▾

As a reminder, all apps created under your account should be related to your business and the type of business it conducts.

By clicking the button below, you agree to [PayPal Developer Agreement](#) (US accounts only).

Create App

***Capture d'écran création app sur le dashboard developper
PayPal***

Les 3 informations sont alors notifiées, et dépendent du compte qui vient d'être créé, comme ci-dessous :

SANDBOX API CREDENTIALS

Sandbox account
sb-evw0115823720@business.example.com

Client ID
AchlSw2ntTA_Nxm91xvvn9W8EDOGzEaiiDxoXXxxSvgjezVly0JC8h0mGXUBBAkKNiNnZTqlyFiGGTHF

Secret
[Hide](#)

Note: When you generate a new secret, you still maintain the original secret. The maximum number of client secrets is two. A client secret is either in enabled or disabled state.

Created	Secret	Status	Action
Apr 19, 2022	EFIJPOLAwxqgMI_NC_YNTUuO_Tv1XRT97rVsdb8BgCz754zUDJ_Ek4By4rCjI 3D0IUgXWqurTearvJoe	Enabled	...

[Generate new secret](#)

Capture d'écran identifiants sur le dashboard developper PayPal

Il s'agit de l'email, du Client ID et du Secret.

Exercice p. 17 Solution n°5

Question 1

Les solutions comme PayPal sont optimisées pour les paiements :

- ☒ Unique
 - ☒ Par abonnement
 - ☒ De test ou SandBox
 - ☐ De Marketplace
- Q** Bien que PayPal puisse être utilisée par une Marketplace, l'API n'est pas optimisé pour.

Question 2

Le développeur peut être contraint sur un API de paiement spécifique par un client.

- ☐ Faux
 - ☒ Vrai
- Vrai, il s'agit de réduire les coûts d'utilisations.*

Question 3

La partie redirect_urls sur les apis permettent :

- ☒ De définir les adresses de redirection d'échec et de succès.
- ☐ De rediriger la transaction vers un autre lien.
- ☒ La partie redirect_urls permet de définir les adresses de redirection d'échec et de succès.

Question 4

Pour les identifiants il est recommandé :

- ☐ D'avoir une unique paire de clé d'application pour toutes les applications.
- ☒ D'avoir une paire de clé d'application spécifique pour chaque application.
- ☒ Il est recommandé que chaque application possède sa paire de clé spécifique afin de limiter les risques de failles.

Question 5

Les points primordiaux d'utilisation d'une API de paiement sont :

- ☒ Sa documentation
- ☒ Son environnement bac à sable
- ☒ La maîtrise du langage de programmation utilisé
- ☐ Le taux de client déjà existant sur la plateforme
- ☒ La documentation, la maîtrise du langage et l'environnement bac à sable sont les points clés d'utilisation d'une API de paiement.