Report – 17/07/2023

Kiet

How to create a meeting using Google Calendar API?

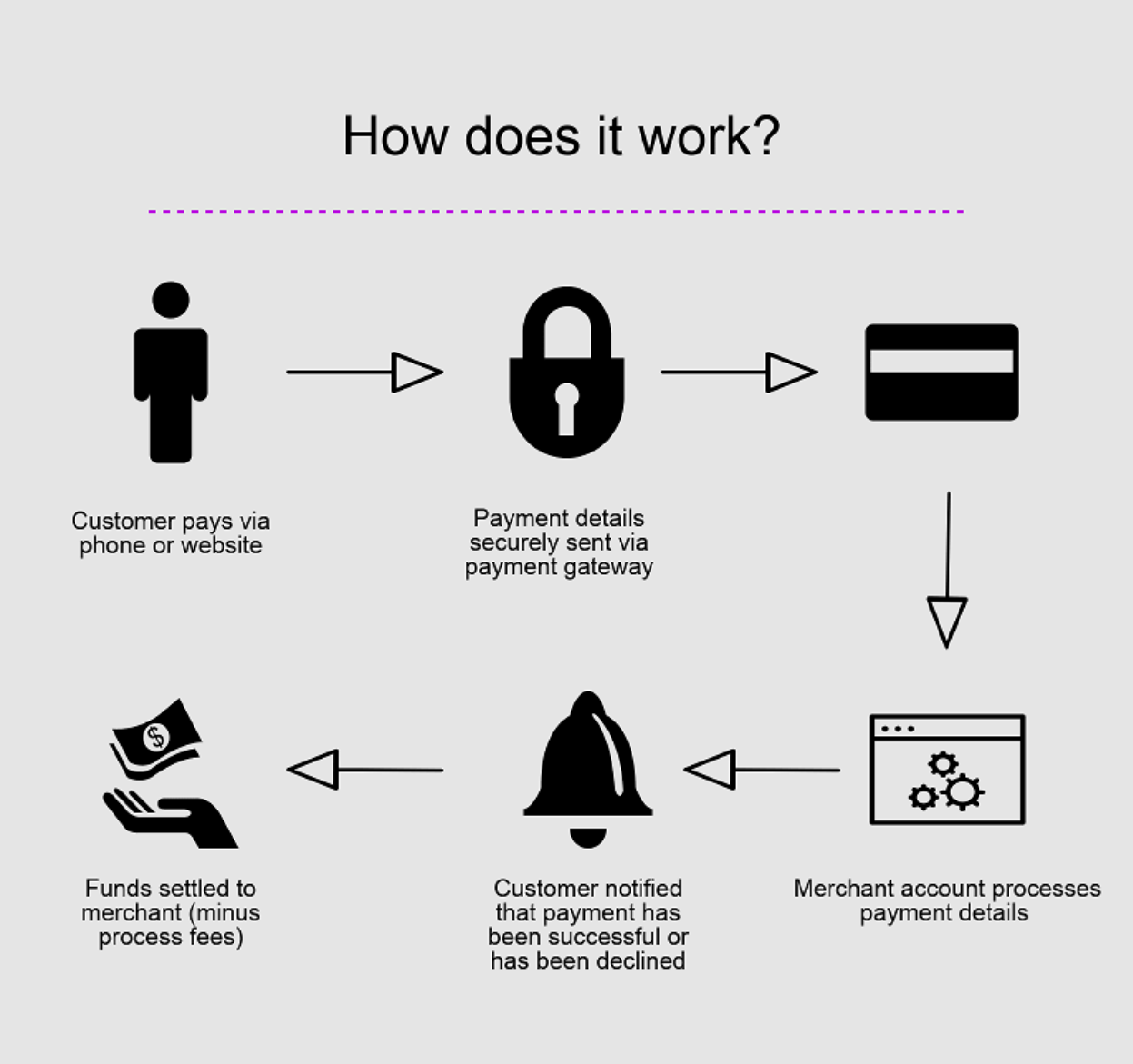
Step 1. Create a Google Cloud Platform project and enable the Google Calendar API.
Step 2. Create an API key and download the JSON file.
Step 3. Add the JSON file to your project.
Step 4. Write code to create a meeting.

Here's an example code to create a meeting using the Google Calendar API:

```
import calendar
import googleapiclient.discovery

# Get your Google Calendar API credentials.
credentials = googleapiclient.discovery.build('calendar',
'v3').get_credentials()

# Create a calendar event.
event = {
  'summary': 'My Meeting',
  'start': {
    'dateTime': '2023-03-08T10:00:00Z'
  },
  'end': {
    'dateTime': '2023-03-08T11:00:00Z'
  },
  'attendees': [
    {
      'email': 'johndoe@example.com'
    },
    {
      'email': 'janedoe@example.com'
    }
  ]
}

# Create the event.
calendar = googleapiclient.discovery.build('calendar', 'v3')
calendar.events().insert(calendarId='primary', body=event,
sendNotifications=True).execute()
```

This code will create a meeting with the title "My Meeting" starting at 10:00 AM and ending at 11:00 AM. The meeting will be scheduled in your primary calendar, and email notifications will be sent to the attendees.

How to integrate payment method into e-commerce website?



PayPal

There is hardly a person today who hasn't heard of PayPal, especially if they are an active user. It is one of the most famous payment systems in the world and, according to Statista, the payment volume in PayPal doubled from 2014 to 2017, and it seems like there is still potential for growth, as the company keeps adding new features to lure customers.

Another advantage of PayPal is that there are many freelancers paid through this system, which accumulates significant sums of money in their accounts. Spending this money within the PayPal ecosystem is easy and convenient, which is an

additional benefit. Providing your potential customers with the possibility of using a system they already know well can bring amazing results.

To pay through PayPal, users don't even have to provide their bank account information. Instead, they just need to create an account within PayPal itself. This platform is especially popular among millennials, so most mid-sized businesses use this type of payment gateway to get paid and increase revenue levels among clients who have these types of accounts. As for large companies, they usually integrate this system as an additional payment gateway.

In this case, the question of how to integrate a payment gateway into a website has a very clear and simple answer. It is being integrated through the Express API, which is a fast and convenient method.

Here are some examples of the Standard PayPal and Express Checkout processing fees:

- No setup or monthly fees
- US fees – 2.9% + $0.30 per transaction.
- International fees – 3.9% + $0.30 per transaction.

PayPal Payments Pro processing fees:

- No setup fees.
- $30 monthly fee.
- US fees – 2.9% + $0.30 per transaction.
- International fees – 3.9% + $0.30 per transaction.
- Virtual terminal fees – 3.1% + $0.30 per domestic transaction; additional 1% for cross-border transactions.

How to integrate PayPal?

You'll need the JavaScript checkout.js script to integrate Express Checkout, and this is provided by PayPal. To render the button on your website, simply add the checkout.js script to your client and also add this code on the page where you want the PayPal button to render:

```
<meta charset="utf-8" />
<meta http-equiv="X-UA-Compatible" content="IE=edge" />
<meta name="viewport" content="width=device-width, initial-scale=1">

<script src="https://www.paypalobjects.com/api/checkout.js"></script>

<div id="paypal-button"> </div>

<script>
   paypal.Button.render({
     env: 'production', // Or 'sandbox',
     commit: true, // Show a 'Pay Now' button
     style: {
       color: 'gold',
       size: 'small'
     },
     payment: function(data, actions) {
       /**Set up the payment here*/
     },
     onAuthorize: function(data, actions) {
   /** Execute the payment here*/
     },
     onCancel: function(data, actions) {
       /**Buyer cancelled the payment*/
     },
     onError: function(err) {
       /**An error occurred during the transaction*/
     }
   }, '#paypal-button');
 </script>

```

With the help of the PayPal checkout script, you can configure the appearance of the button on your website and define the scenario after buyer authorizes or cancels the payment, as well as determine what happens in case of an error.

- #paypal-button - Container element in which the button is rendered.
- Style - Button appearance customization.
- payment() - Function called at a time when your customer clicks the button. Here you can set up and return a payment to start the checkout process.
- onAuthorize() - Function called when your customer logs in and authorizes the payment. Here you may show a confirmation page and take the payment.
- onCancel() - Function called in case of payment cancellation. By default, the customer is taken back to the original page; however, you can change this setting and take them to any other page.
- onError() - Function called in case of an error. Here you may let your customer try again or simply show an error notification.

- #paypal-button - Container element in which the button is rendered.

PayPal SDK:
https://developer.paypal.com/sdk/js/

Ref:
https://dinarys.com/blog/integrate-payment-gatewa-in-your-ecommerce-website

https://developers.google.com/calendar/