

# API REST

## ¿Qué es una REST API?

Es una interfaz que permite la comunicación entre sistemas a través de peticiones HTTP. REST es un estilo arquitectónico para diseñar servicios web, y una API basada en REST sigue los principios REST para facilitar la interacción con recursos (como datos) que están representados a través de URLs.

### Características clave de una REST API:

- **Stateless** (Sin estado): Cada petición es independiente y no guarda información sobre peticiones anteriores.
- **Recursos**: Los recursos, como usuarios o productos, se identifican mediante URLs.
- **Representaciones**: Los recursos se representan en varios formatos, como JSON o XML.
- **Verbos HTTP**: Utiliza métodos HTTP (GET, POST, PUT, DELETE, etc.) para interactuar con los recursos.

REST APIs son comunes en el desarrollo web moderno para crear servicios que permiten la comunicación entre aplicaciones, como la interacción entre un frontend y un backend.

## ¿Qué es un END POINT?

Es una URL específica en una API que responde a una solicitud concreta. Representa una ruta donde se pueden realizar operaciones sobre recursos mediante métodos HTTP.

Por ejemplo:

- En una API de usuarios, un **endpoint** para obtener todos los usuarios podría ser <https://api.example.com/users>, mientras que un **endpoint** para obtener información de un usuario específico sería <https://api.example.com/users/123>.
- Cada endpoint está asociado con una acción particular, como obtener datos, actualizar información o eliminar un recurso.

En resumen, un **endpoint** es la dirección o punto de acceso para interactuar con un recurso a través de una API.

## ¿Qué métodos HTTP se utilizan en una REST API?

Los métodos HTTP, también conocidos como **verbos HTTP**, indican la acción que deseas realizar en el recurso. Los más comunes en una REST API son:

1. **GET**: Obtiene información sobre un recurso.
  - Ejemplo: Obtener una lista de productos (**GET /products**).
2. **POST**: Crea un nuevo recurso.
  - Ejemplo: Crear un nuevo usuario (**POST /users**).
3. **PUT**: Actualiza un recurso existente (puede reemplazar todos los datos del recurso).
  - Ejemplo: Actualizar toda la información de un producto (**PUT /products/123**).
4. **PATCH**: Actualiza parcialmente un recurso (modifica solo algunas propiedades).
  - Ejemplo: Actualizar solo el precio de un producto (**PATCH /products/123**).
5. **DELETE**: Elimina un recurso.
  - Ejemplo: Eliminar un usuario (**DELETE /users/123**).

### Otros métodos:

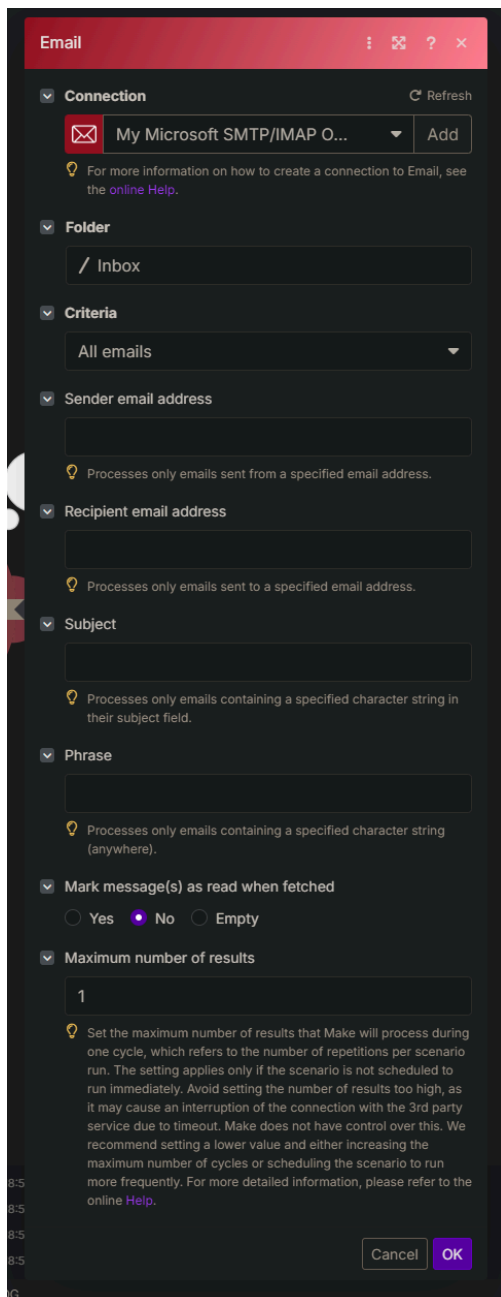
- **HEAD**: Igual que **GET** pero sin el cuerpo de la respuesta, solo para obtener los metadatos.
- **OPTIONS**: Muestra las opciones de comunicación disponibles para el recurso.

Estos métodos definen las operaciones que puedes realizar sobre los recursos dentro de una REST API.

**Propón una automatización en MAKE utilizando 2/3 servicios:**  
<https://www.make.com/en/integrations?community=1&verified=1>

### EJEMPLO 01

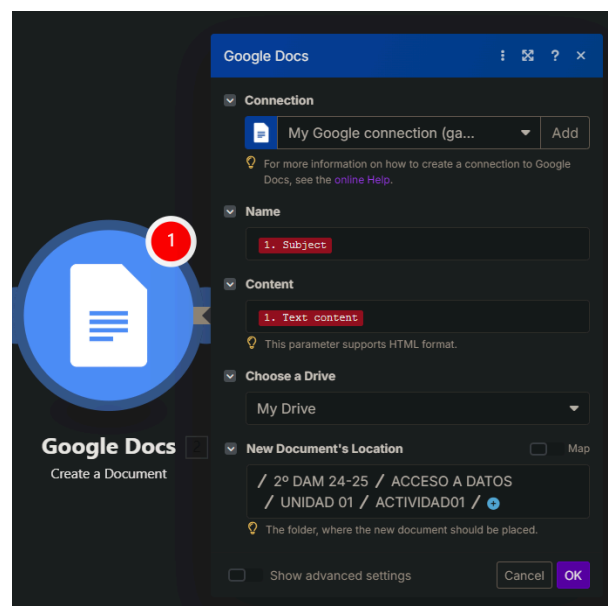
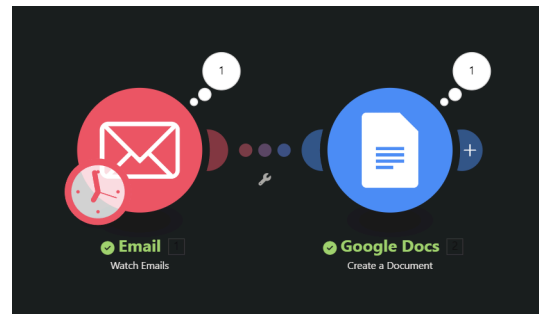
Empleo los servicios de Email y Google Docs para crear un documento a partir del primero correo electrónico.



The screenshot shows the 'Email' configuration window in Make.com. It has a red header bar with the title 'Email' and icons for settings, help, and close. The configuration is divided into several sections, each with a checkbox and a title:

- Connection:** Includes a dropdown menu showing 'My Microsoft SMTP/IMAP O...' and an 'Add' button. A help icon and text state: 'For more information on how to create a connection to Email, see the online Help.'
- Folder:** A text input field containing '/ Inbox'.
- Criteria:** A dropdown menu set to 'All emails'.
- Sender email address:** An empty text input field. A help icon and text state: 'Processes only emails sent from a specified email address.'
- Recipient email address:** An empty text input field. A help icon and text state: 'Processes only emails sent to a specified email address.'
- Subject:** An empty text input field. A help icon and text state: 'Processes only emails containing a specified character string in their subject field.'
- Phrase:** An empty text input field. A help icon and text state: 'Processes only emails containing a specified character string (anywhere).'
- Mark message(s) as read when fetched:** Three radio buttons: 'Yes', 'No' (selected), and 'Empty'.
- Maximum number of results:** A text input field containing '1'. A help icon and text state: 'Set the maximum number of results that Make will process during one cycle, which refers to the number of repetitions per scenario run. The setting applies only if the scenario is not scheduled to run immediately. Avoid setting the number of results too high, as it may cause an interruption of the connection with the 3rd party service due to timeout. Make does not have control over this. We recommend setting a lower value and either increasing the maximum number of cycles or scheduling the scenario to run more frequently. For more detailed information, please refer to the online Help.'

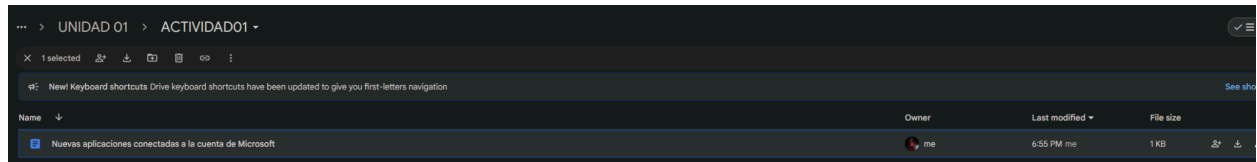
At the bottom right, there are 'Cancel' and 'OK' buttons.



The screenshot shows the 'Google Docs' configuration window in Make.com. It has a blue header bar with the title 'Google Docs' and icons for settings, help, and close. The configuration is divided into several sections, each with a checkbox and a title:

- Connection:** Includes a dropdown menu showing 'My Google connection (ga...' and an 'Add' button. A help icon and text state: 'For more information on how to create a connection to Google Docs, see the online Help.'
- Name:** A text input field containing '1. Subject'.
- Content:** A text input field containing '1. Text content'. A help icon and text state: 'This parameter supports HTML format.'
- Choose a Drive:** A dropdown menu set to 'My Drive'.
- New Document's Location:** A text input field containing '/ 2º DAM 24-25 / ACCESO A DATOS / UNIDAD 01 / ACTIVIDAD01 /'. A help icon and text state: 'The folder, where the new document should be placed.'

At the bottom right, there are 'Show advanced settings', 'Cancel', and 'OK' buttons.

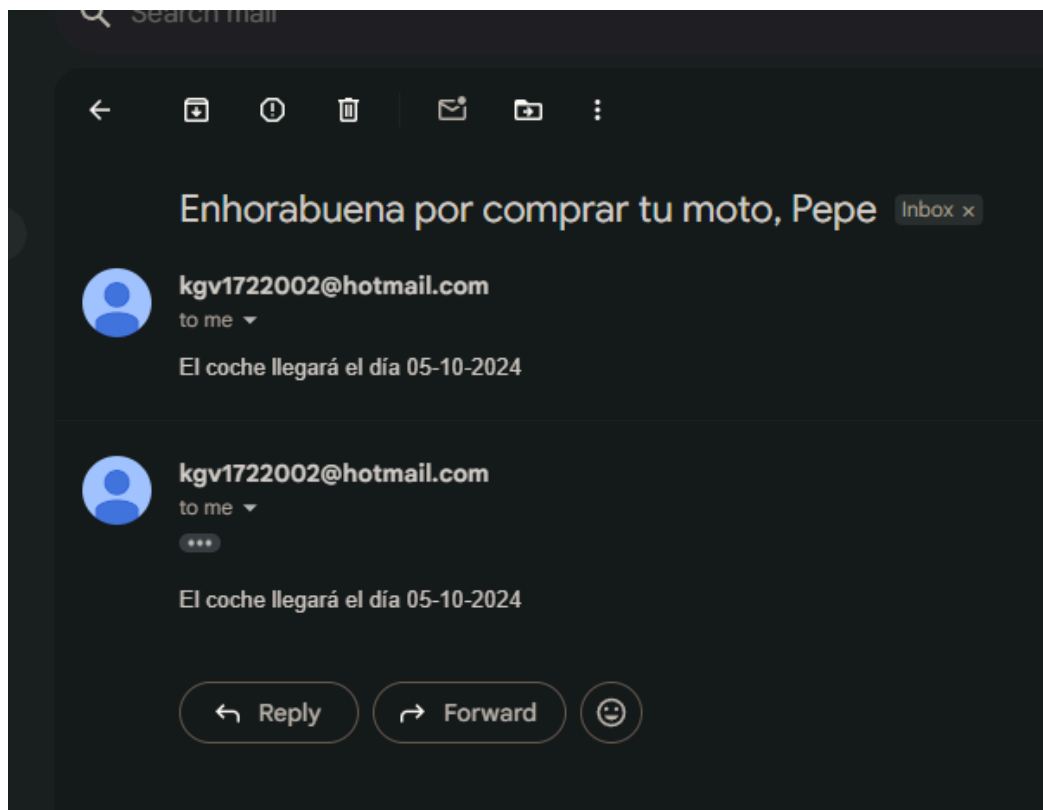
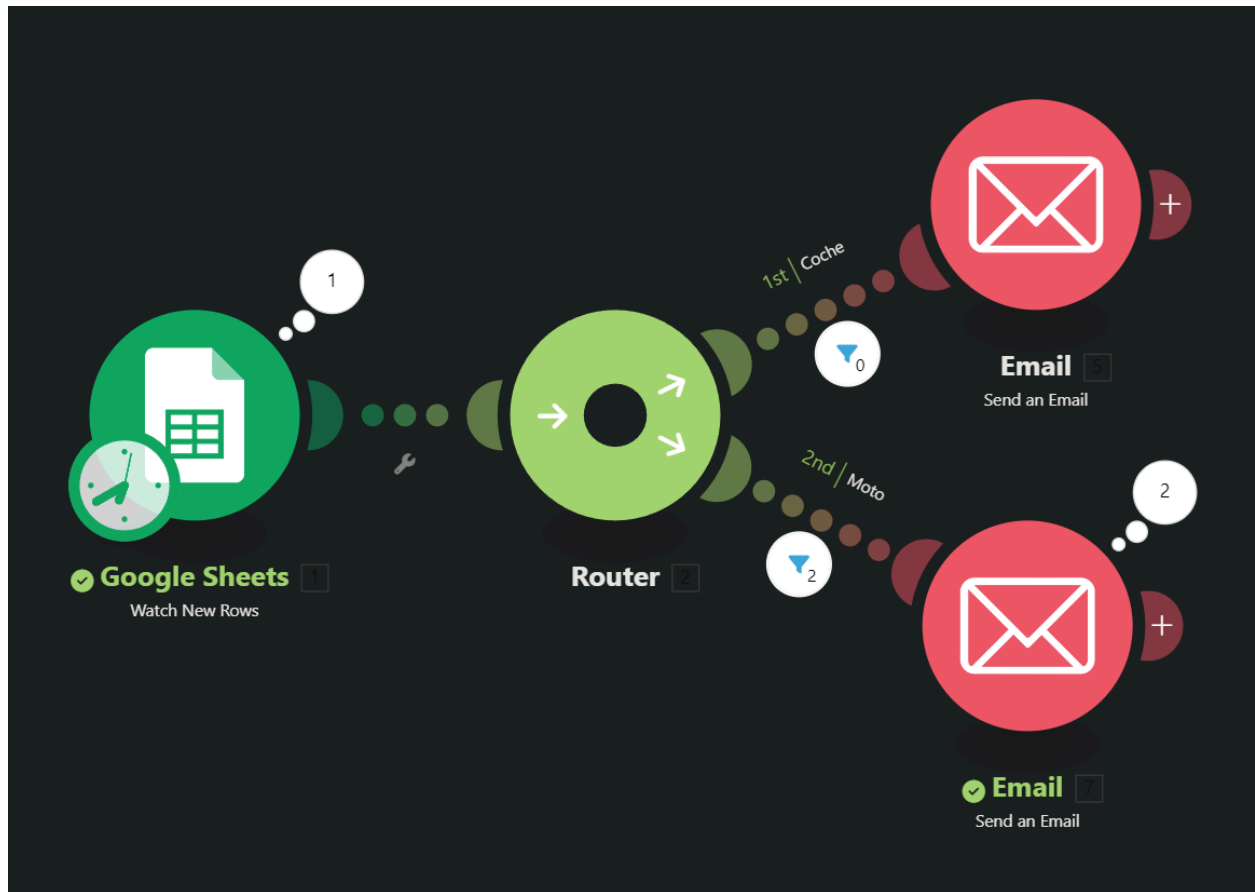


## EJEMPLO 02

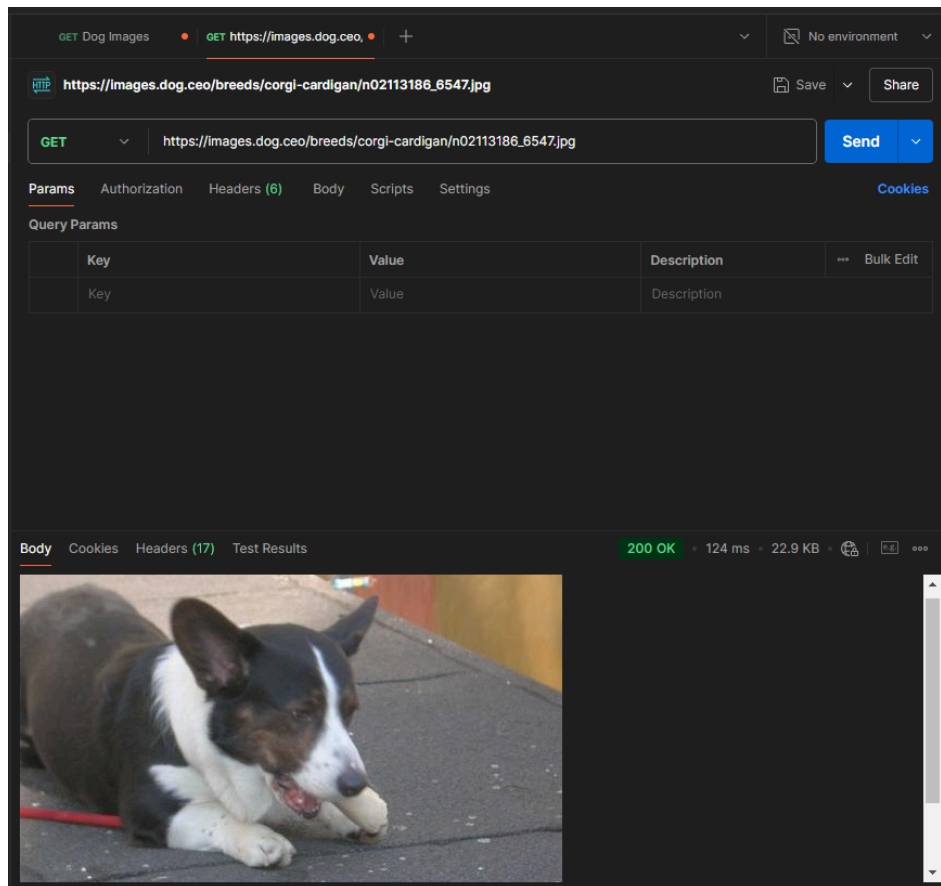
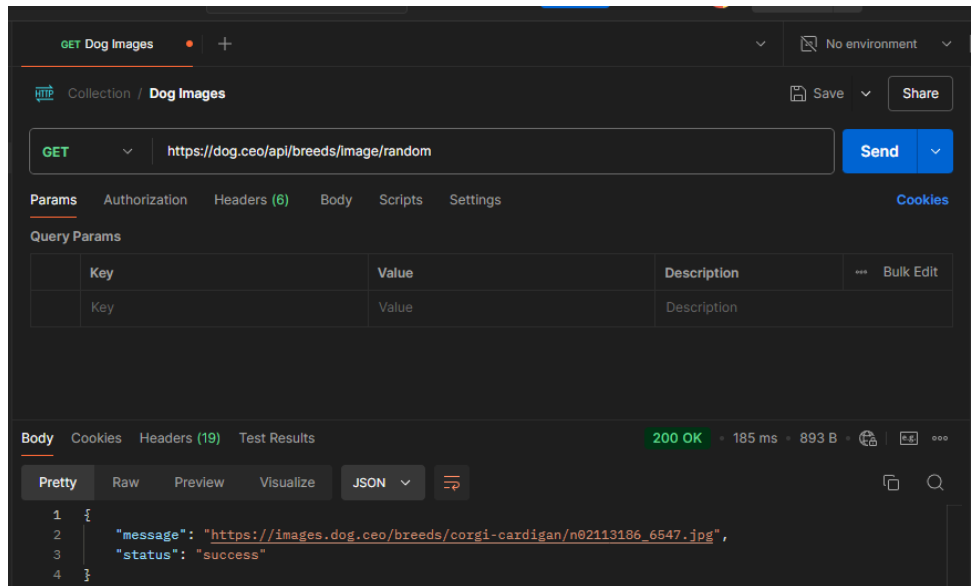
Empleo los servicios de Google Sheet, Email y un Router para la gestión de un concesionario. Envía dos correos electrónicos a partir de los datos del Google Sheet, que muestra la fecha la cual se entregará el vehículo comprado siempre y cuando el precio sea el indicado.



| Nombre |                          |        |   |   |   |               |                |
|--------|--------------------------|--------|---|---|---|---------------|----------------|
| A      | B                        | C      | D | E | F | G             | H              |
| Nombre | Email                    | Precio |   |   |   | Precios Motos | Precios Coches |
| tino   | pruebas1722002@gmail.com | 10000  |   |   |   | 1000€ a 1200€ | 6000€ a 10000€ |
| Alvaro | pruebas1722002@gmail.com | 7000   |   |   |   |               |                |
| PEPE   | pruebas1722002@gmail.com | 1200   |   |   |   |               |                |
| PEPE   | pruebas1722002@gmail.com | 1200   |   |   |   |               |                |
| tino   | pruebas1722002@gmail.com | 10000  |   |   |   |               |                |
| Alvaro | pruebas1722002@gmail.com | 7000   |   |   |   |               |                |
| PEPE   | pruebas1722002@gmail.com | 1200   |   |   |   |               |                |



**Instala POSTMAN en tu equipo**  
**Realiza varias llamadas a los END POINT de <https://dog.ceo/> y adjunta las capturas.**



## **Conclusiones: Realiza una conclusión de lo que has aprendido en la práctica.**

Durante esta práctica, he aprendido los conceptos fundamentales y cómo interactuar con APIs REST. He comprendido que una REST API es una forma eficiente de permitir la comunicación entre sistemas utilizando peticiones HTTP, siguiendo un enfoque sin estado (stateless) y basándose en recursos accesibles mediante URLs. Además, me ha quedado claro el uso de verbos HTTP (GET, POST, PUT, DELETE, PATCH) y cómo estos permiten realizar diferentes operaciones sobre los recursos, como obtener, crear, actualizar y eliminar datos.

El uso de Postman me ha permitido comprender cómo realizar llamadas a endpoints específicos de una API para probar y verificar su funcionamiento. Además, con la práctica de realizar varias llamadas a <https://dog.ceo/>, he podido ver cómo se gestionan las peticiones y respuestas en una API pública.

Finalmente, también aprendí cómo integrar servicios en Make para automatizar tareas, como la creación de documentos a partir de correos electrónicos o la gestión de información en Google Sheets para enviar correos automatizados. Esta parte de la práctica me ayudó a entender cómo se puede combinar la funcionalidad de diferentes aplicaciones para resolver problemas específicos y automatizar procesos.

En general, la práctica me brindó una buena base sobre APIs REST y cómo utilizarlas en la comunicación entre sistemas y la automatización de tareas en entornos de desarrollo web.