



# EXCHANGEHUB

## DOCUMENTACIÓN DE LA API DE EXCHANGEHUB

JULIAN ESTIVEN POSSO CATAÑO, GERONIMO TRUJILLO, JUAN RUIZ

Centro de la Manufactura Avanzada

## **Introducción**

En el desarrollo de aplicaciones web modernas, las API (Interfaces de Programación de Aplicaciones) juegan un papel fundamental al permitir la comunicación entre el frontend y el backend. En este documento, se presenta la documentación de la API de ExchangeHub, una plataforma diseñada para facilitar el intercambio de artículos entre usuarios.

La documentación sigue el estándar OpenAPI 3.0 y se ha generado con Swagger para ofrecer una referencia clara y estructurada sobre los endpoints disponibles, los métodos HTTP utilizados y los parámetros requeridos. Esto permite a los desarrolladores integrar y consumir la API de manera eficiente, asegurando un uso adecuado de sus funcionalidades.

A lo largo del documento, se detallan los endpoints disponibles para la autenticación de usuarios, la gestión de artículos, las solicitudes de intercambio y la administración de perfiles. Cada sección incluye ejemplos de uso y posibles respuestas del servidor, facilitando su implementación en diversas aplicaciones.

Esta documentación está estructurada conforme a las normas de la American Psychological Association (APA), garantizando la correcta citación y presentación de la información técnica de la API.

POST	/register	Registra un nuevo usuario	🔗
POST	/login	Inicia sesión en la plataforma	⌵
POST	/logout	Cierra la sesión del usuario	⌵
GET	/verify	Verifica si el usuario tiene sesión activa	⌵
GET	/articles	Obtiene todos los artículos disponibles	⌵
POST	/articles	Crea un nuevo artículo	⌵
GET	/articles/user/{id}	Obtiene los artículos de un usuario específico	⌵
POST	/articles/image	Sube una imagen de un artículo	⌵
GET	/articles/image/{id}	Obtiene imágenes de un artículo	⌵
DELETE	/articles/{id}	Elimina un artículo	⌵
GET	/articles/search/{search}	Busca artículos por nombre o descripción	⌵
GET	/articles/category/{category}	Filtra artículos por categoría	⌵
POST	/exchanges	Crea una nueva solicitud de intercambio	⌵
PATCH	/exchanges/cancel/{id}	Cancela un intercambio	⌵
PATCH	/users/{iduser}/image	Sube una imagen de perfil	⌵
PUT	/users/{iduser}	Actualiza la información del usuario	⌵

## Post/register (Registra un nuevo usuario)

“name, lastname, email, address, cellphone, password ”

POST

/register

Registra un nuevo usuario

⌵

Parameters

Try it out

No parameters

Request body

required

application/json

⌵

Example Value

Schema

```
{
  "name": "string",
  "lastname": "string",
  "email": "string",
  "address": "string",
  "cellphone": 0,
  "password": "string"
}
```

Responses

Code	Description	Links
201	Usuario registrado con éxito	No links

## Post/login (Inicia sesión en la plataforma)

“email, password”

POST

/login

Inicia sesión en la plataforma

^

Parameters

Try it out

No parameters

Request body required

application/json

Example Value | Schema

```
{  "email": "string",  "password": "string"}}
```

Responses

Code	Description	Links
200	Inicio de sesión exitoso	No links

## Post/logout (cierra la sesión del usuario)

“no parameters”

POST

/logout

Cierra la sesión del usuario

^

Parameters

Try it out

No parameters

Responses

Code	Description	Links
200	Sesión cerrada exitosamente	No links

Get/verify (verifica si el usuario tiene sección activa)

“no parameters”

GET /verify Verifica si el usuario tiene sesión activa ^		
Parameters Try it out		
No parameters		
Responses		
Code	Description	Links
200	Usuario verificado	No links

Get/articles (obtiene todos los artículos disponibles)

“no parameters”

GET /articles Obtiene todos los artículos disponibles ^		
Parameters Try it out		
No parameters		
Responses		
Code	Description	Links
200	Lista de artículos	No links

Post/articles (crea un nuevo artículo)

“nombre, descripción, categoría, imagen”

**POST** /articles Crea un nuevo artículo

Parameters Try it out

No parameters

Request body required application/json

Example Value | Schema

```
{
  "nombre": "string",
  "descripcion": "string",
  "categoria": "string",
  "imagen": "string"
}
```

Responses

Code	Description	Links
201	Artículo creado con éxito	No links

Get/articles/user/{id} (obtiene todos los artículos de un usuario específico)

“id”

**GET** /articles/user/{id} Obtiene los artículos de un usuario específico

Parameters Try it out

Name	Description
<b>id</b> <small>* required</small> integer (path)	<input type="text" value="id"/>

Responses

Code	Description	Links
200	Lista de artículos del usuario	No links

Post/articles/image (sube una imagen de un artículo)

“image”

**POST** /articles/image Sube una imagen de un artículo

Parameters Try it out

No parameters

Request body required multipart/form-data

image

string(\$binary)

Responses

Code	Description	Links
201	Imagen subida con éxito	No links

Get/articles/image/{id} obtiene imagenes de un articulo

“id”

**GET** /articles/image/{id} Obtiene imágenes de un artículo

Parameters Try it out

Name	Description
<b>id</b> <small>* required</small> integer (path)	<div>id</div>

Responses

Code	Description	Links
200	Imagen del artículo obtenida	No links

Delete/articles/{id} (elimina un artículo)

“id”

DELETE

/articles/{id} Elimina un artículo

^

Parameters

Try it out

Name	Description
id * required	
integer	
(path)	

Responses

Code	Description	Links
200	Artículo eliminado con éxito	No links

Get/articles/search/{search}

“search(búsqueda del usuario)”

GET

/articles/search/{search} Busca artículos por nombre o descripción

| ^

Parameters

Try it out

Name	Description
search * required	
string	
(path)	

Responses

Code	Description	Links
200	Lista de artículos encontrados	No links

Get/articles/category/{category}



“category”

GET

/articles/category/{category}

Filtra artículos por categoría

^

Parameters

Try it out

Name	Description
category <sup>required</sup>	
string (path)	<input type="text" value="category"/>

Responses

Code	Description	Links
200	Lista de artículos filtrados por categoría	No links

Post/exchanges (Crea una nueva solicitud de intercambio)

“No Parameters”

Request Body “ProductOne, ProductTwo, userOne, userTwo”

POST

/exchanges

Crea una nueva solicitud de intercambio

^

Parameters

Try it out

No parameters

Request body <sup>required</sup>

application/json

Example Value | Schema

```
{  "productOne": 0,  "productTwo": 0,  "userOne": 0,  "userTwo": 0}
```

Responses

Code	Description	Links
201	Intercambio creado con éxito	No links

Patch/exchanges/cancel/{id} (cancela in intercambio)

“id”

Request body “status”

PATCH /exchanges/cancel/{id} Cancela un intercambio

Try it out

Parameters

Name	Description
id <small>required</small>	
integer	
(path)	

Request body required

application/json

Example Value | Schema

```
{  
  "status": "string"  
}
```

Responses

Code	Description	Links
200	Intercambio cancelado	No links

Patch/users/{idUser}/image (sube una imagen de perfil)

“idUser”

Request body “image”

PATCH /users/{idUser}/image Sube una imagen de perfil

Try it out

Parameters

Name	Description
idUser <small>required</small>	
integer	
(path)	

Request body required

multipart/form-data

image

string(binary)

Responses

Code	Description	Links
200	Imagen de perfil actualizada	No links

Put/users/{iduser} (actualiza la información del usuario)

“iduser”

Request body “nombre, email”

PUT

/users/{iduser}

Actualiza la información del usuario

^

Parameters

Try it out

Name	Description
<b>iduser</b> <small>* required</small>	
integer	
(path)	iduser

Request body required

application/json

Example Value

Schema

```
{
  "nombre": "string",
  "email": "string"
}
```

Responses

Code	Description	Links
200	Información del usuario actualizada	No links