

Tema 7

Microservicios y APIs

Módulo de Desarrollo de Interfaces

2º DAM

Objetivos

- Arquitectura Cliente-Servidor
- Conocer el concepto de microservicios.
- APIs
 - Protocolo HTTP
 - API RestFull
- Ejemplos de desarrollo

Arquitectura de aplicaciones

Módulo de Desarrollo de Interfaces

2º DAM

Arquitectura

La arquitectura de una aplicación es el diseño estructural que organiza sus componentes y define cómo interactúan entre sí.

- Favorece la **escalabilidad, mantenibilidad y rendimiento.**
- Influye en la **experiencia de usuario** y el desarrollo futuro.

Componentes principales

- **Presentación:** Interfaz con el usuario (UI/UX).
- **Lógica de negocio:** Procesa reglas y operaciones.
- **Datos:** Gestión de la información y su almacenamiento.

Tipos comunes de arquitectura

Monolítica:

- Todo integrado en un único bloque.
- Simplicidad inicial pero difícil de escalar y mantener.

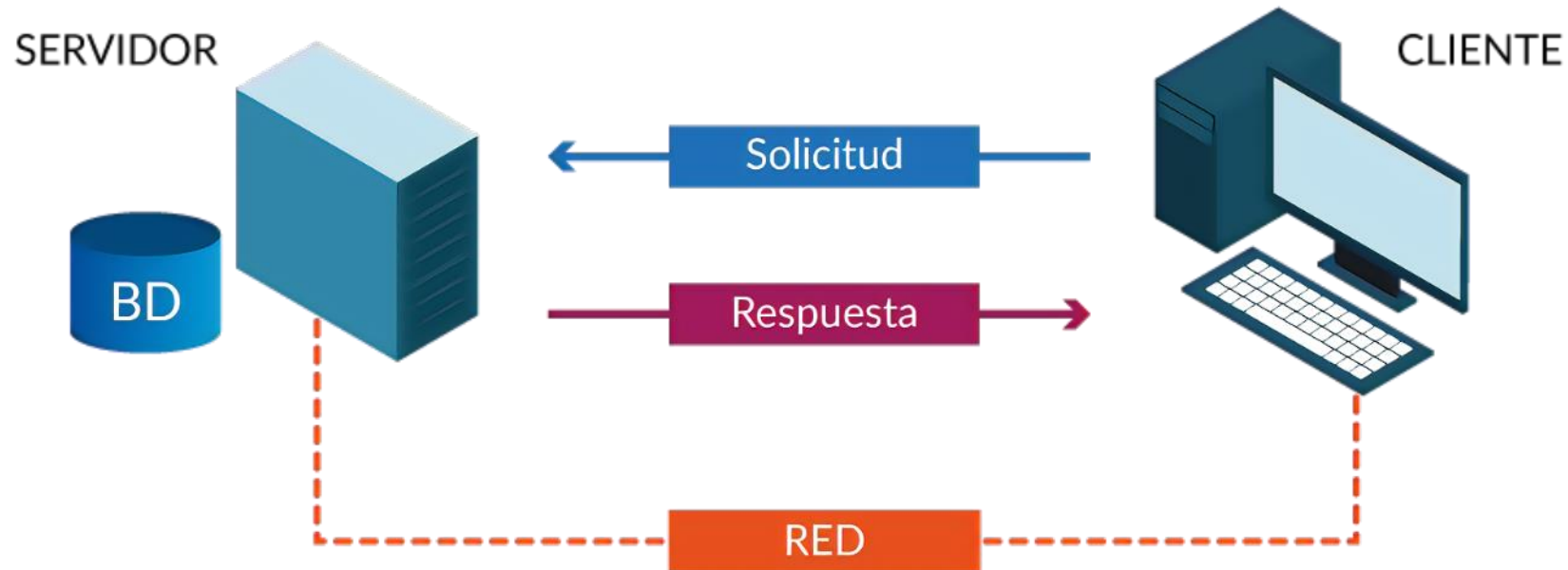
Cliente-Servidor:

- División en cliente (interfaz) y servidor (lógica y datos).
- Separación de funciones pero depende de una buena comunicación entre ambos.

Microservicios:

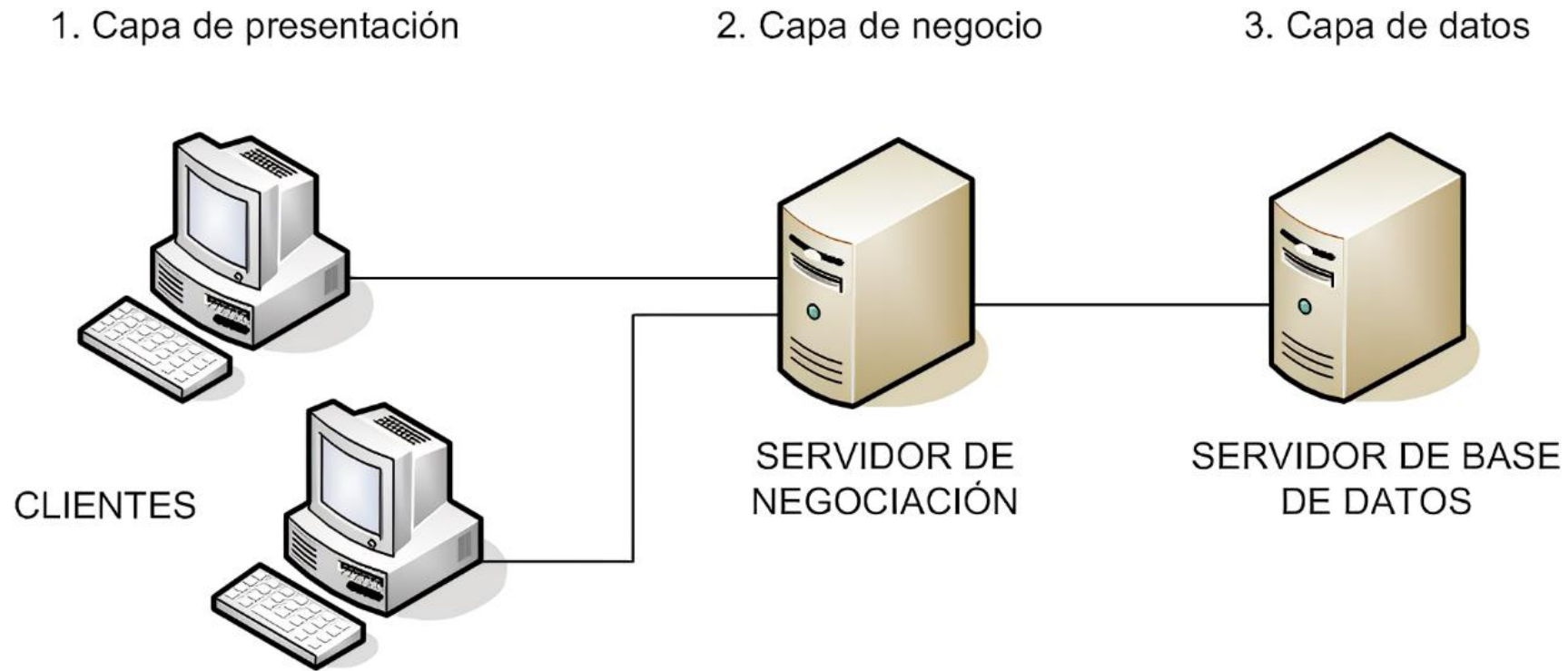
- Servicios pequeños e independientes.
- Mucha escalabilidad y flexibilidad a costa de mayor complejidad de gestión.

Cliente Servidor



En este tipo de arquitectura un **cliente** que usa los servicios de un **servidor** donde se ejecuta **todo lo necesario** para poder dar servicio a las aplicaciones.

Distribución por capas



Microservicios

Módulo de Desarrollo de Interfaces

2º DAM

Microservicios

- Los microservicios son un patrón de diseño software a nivel arquitectónico que implementa servicios mediante la colaboración otros servicios más pequeños y autónomos.
- Cada microservicio debe dar solución a un área de negocio concreta, abstrayendo al resto del sistema de los detalles concretos de la implementación, favoreciendo su independencia, el mantenimiento y la evolución de cada uno de ellos.

Ventajas

- La escalabilidad es más eficiente.
- Potencia la diversidad tecnológica: utilizar múltiples lenguajes así como diferentes stacks tecnológicos.
- Permite que los desarrollos sean independientes y en paralelo.
- Facilita el mantenimiento.
- Permite despliegues independientes.
- Aumenta la tolerancia a fallos.

Inconvenientes

- Aumentará la complejidad en la gestión de la configuración. Cada microservicio necesitará su propio canal de construcción y despliegue.
- También aumentará la complejidad para mantener la transaccionalidad de cada operación.
- El rendimiento se puede ver afectado debido a saturaciones de la red o a procesos de (de)serialización.
- Monitorizar el sistema será más complejo.

Ejemplo: tienda online

- **Catálogo de Productos:** Gestiona y muestra los productos.
 - **Carrito de Compras:** Permite agregar y gestionar productos seleccionados.
 - **Procesamiento de Pagos:** Valida pagos y genera facturas.
 - **Gestión de Usuarios:** Controla los perfiles y autenticación.
-
- ✓ Podemos escalar solo el microservicio de pagos si la demanda aumenta.
 - ✓ Actualizar el carrito no afecta al catálogo.
 - ✓ Si el servicio de usuarios falla, el resto puede seguir funcionando.

APIs

Módulo de Desarrollo de Interfaces

2º DAM

Qué es una API

API

Application Programming Interface

Una API o interfaz de programación de aplicaciones es un conjunto de definiciones y protocolos que se usa para diseñar e integrar el software de las aplicaciones.

Qué es una API

- Las API permiten que sus productos y servicios se comuniquen con otros, sin necesidad de saber cómo están implementados.
- Esto simplifica el desarrollo de las aplicaciones y permite ahorrar tiempo y dinero.
- Las API otorgan flexibilidad; simplifican el diseño, la administración y el uso de las aplicaciones.
- Ofrecen oportunidades de innovación, lo cual es ideal al momento de diseñar herramientas y productos nuevos.

APIs Remotas

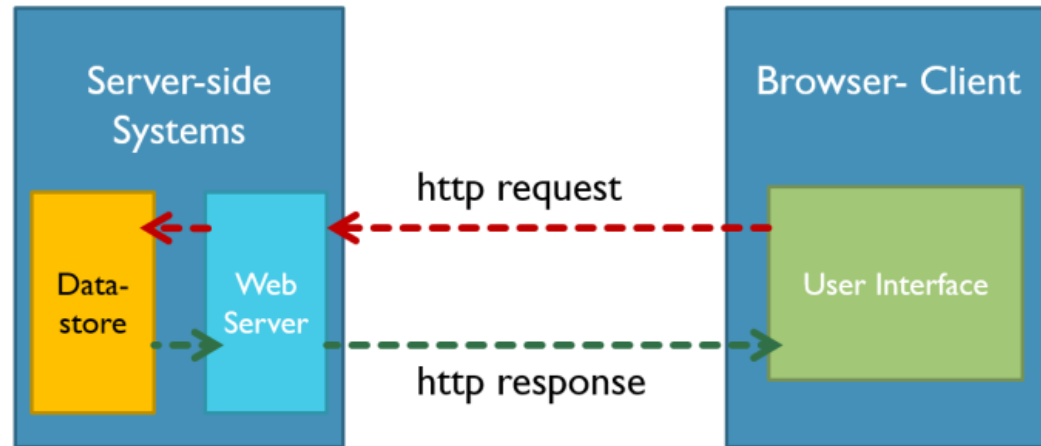
- Las API remotas están diseñadas para interactuar en una red de comunicaciones.
- Los recursos que administra la API se encuentran fuera de la máquina que envía la solicitud.
- La mayoría de las API están diseñadas de acuerdo con los estándares web.
- Por lo general, las API web usan HTTP para los mensajes de solicitud.
- Los mensajes de respuesta suelen ser archivos XML o JSON.

REST

Representational State Transfer

Es un patrón de diseño (no es una especificación) para la creación de APIs remotas que utilizan HTTP como protocolo de comunicación.

API Rest



▼ General	
URL de la solicitud:	https://api.wallapop.com/api/v3/general/search?keywords=playstation%205.&latitude=40. arch_box&order_by=closest
Método de la solicitud:	OPTIONS
Código de estado:	✔ 200 OK
Dirección remota:	18.202.73.16:443
Directiva de sitio de referencia:	strict-origin-when-cross-origin
▼ Encabezados de respuesta Ver origen	
Access-Control-Allow-Credentials:	true
Access-Control-Allow-Headers:	deviceos, x-deviceos
Access-Control-Allow-Methods:	GET,HEAD,POST
Access-Control-Allow-Origin:	https://es.wallapop.com

API Rest

- En una API Rest se modelan recursos a los que se acceden a través de una URI (Uniform Resource Identifier).
- Se usan los verbos propios del protocolo HTTP:
 - GET
 - POST
 - PUT
 - DELETE

Programación de APIs



Flask

Ejemplo y uso de APIs

{JSON} Placeholder

Free fake API for testing and prototyping.

Powered by [JSON Server](#) + [LowDB](#)

As of Oct 2021, **serving ~1.7 billion requests each month.**

Elementos de una API REST

- Recurso
- EndPoint
- Ruta de acceso al recurso
- Petición (cabecera y cuerpo)
- Método de acceso
- Respuesta

Ejemplos de end-point

- /post --> para obtener información de un post
- /posts --> para obtener todos los post
- /user --> para obtener información de un usuario
- /users --> para obtener todos los usuarios

Métodos de acceso

- Los principales son GET y POST
 - **Get** permite pasar parámetros a través de la url, por lo que los parámetros son visibles. Solo se permiten parámetros sencillos.
 - **Post** permite pasar parámetros a través de la cabecera de la petición, por lo que no son visibles. Se permiten parámetros complejos (arrays, archivos, etc...)
- **SE PUEDEN COMBINAR AMBOS**

Paso de parámetros a la api

- A traves del propio end-point
 - `/post/{id}` --> `/post/4` --> devuelve el post id=4
- A traves de la url (get)
 - `/post?id=4` `/equipo.jsp?nombre=barcelona`
- A traves de la petición http (post)
 - `/post` `post(id=4)`

Ejemplo con SpringBoot

```
@GetMapping("/empleado/{id}")
Empleado obtenerEmpleado(@PathVariable Long id) {
    return empleadosRepository.findById(id);
}
```

Ejemplo con Express

```
app.get('/empleado/:id', (req, res) => {  
  const id = parseInt(req.params.id, 10);  
  const empleado = empleadosRepository.findById(id);  
  res.json(empleado);  
});
```

Ejemplo con Flask

```
@app.route('/post/<int:post_id>')  
def show_post(post_id):  
    return f'Post {post_id}'
```


Respuesta

- La respuesta suele ser JSON o bien un html, aunque puede ser de cualquier tipo válido (csv, pdf, txt, ...)
- Cuando la respuesta es HTML, se puede mostrar en el navegador y tiene como ventaja que es directamente legible por los usuarios.
- En ese caso se suelen usar plantillas para dar formato a la respuesta.



Motores de plantillas HTML

- Python --> Jinja2
- Javascript, Node.js --> Moustache
- PHP --> Twig, Smarty
- Spring Java --> Thymeleaf

Ejemplo de API pública

[Discover](#)[Earn Points](#)[Support](#)

Search TheTVDB



[Home](#) / [Developers](#) /

API and Data Licensing

Thousands of projects, companies, and individuals make use of the data that is available in our API, from large-scale commercial projects/companies to individual end users who contribute data to the site to power their media centers and applications. Our API is available for both commercial projects and to individual developers.

[Pricing](#)[Documentation & Assistance](#)[Attribution](#)[Legal Usage](#)

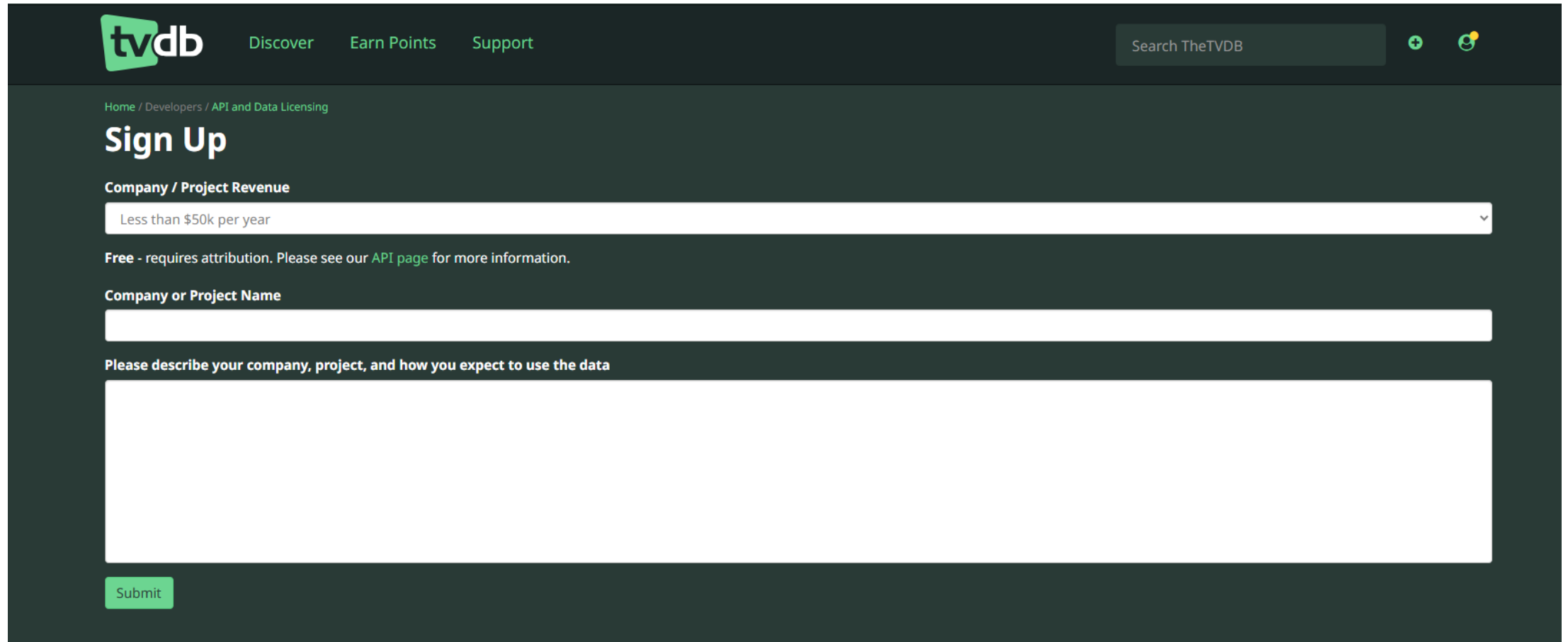
TheTVDB relies on contributed information from tens of thousands of users, many using free and open source software that uses our API. In order to ensure the ongoing quality of our data, we offer significant discounts for these projects.

When determining your tier, please consider the revenue of your parent company. It is your responsibility to ensure that you select the correct tier, and make annual adjustments as necessary.

Company Revenue	Licensing Fee
Less than \$50k per year	free requires attribution
\$50k to \$250k per year	\$1,000 / year
\$250k to \$1M per year	\$10,000 / year
\$1M+ or custom terms	Contact Us

Get Started

Ejemplo de API pública



The screenshot shows the 'Sign Up' page for TheTVDB's public API. The page has a dark theme with green accents. At the top, there's a navigation bar with the TVDB logo, links for 'Discover', 'Earn Points', and 'Support', a search bar, and user icons. The breadcrumb trail is 'Home / Developers / API and Data Licensing'. The main heading is 'Sign Up'. Below it, the section 'Company / Project Revenue' has a dropdown menu currently set to 'Less than \$50k per year'. A note states 'Free - requires attribution. Please see our API page for more information.' The 'Company or Project Name' field is an empty text input. The 'Please describe your company, project, and how you expect to use the data' field is a large, empty text area. A green 'Submit' button is at the bottom left.

tvdb Discover Earn Points Support Search TheTVDB

Home / Developers / API and Data Licensing

Sign Up

Company / Project Revenue

Less than \$50k per year

Free - requires attribution. Please see our [API page](#) for more information.

Company or Project Name

Please describe your company, project, and how you expect to use the data

Submit

Ejemplo de API pública

TheTVDB

Dashboard Home

Account

Edit Information

Change password

Subscription

API keys

Notifications 1

Your Artwork

Favorites

Lists

API Keys

API keys are now per-project and not for use by individual users. If you are a developer, you can sign up for an API key on our [API page](#). If you are an individual user, you shouldn't need an API key. You can consider

Your API Keys


Key	Project Name	Funding Model
8afd88e1-0b70-4e32-a333-9e8cf66df47f	test cesur	Negotiated Contract

Your Legacy Keys

Legacy API keys are deprecated and will be discontinued soon. Developers should update their projects to make use of our new API.

You have no legacy API keys.

Ejemplo de API pública

 **Swagger**
Supported by SMARTBEAR

swagger.yml

Explore

TVDB API V4 4.7.10 OAS3

[swagger.yml](#)

Documentation of [TheTVDB](#) API V4. All related information is linked from our [Github repo](#). You might also want to use our [Postman collection] (<https://www.getpostman.com/collections/7a9397ce69ff246f74d0>)

Authentication

1. Use the /login endpoint and provide your API key as "apikey". If you have a user-supported key, also provide your subscriber PIN as "pin". Otherwise completely remove "pin" from your call.
2. Executing this call will provide you with a bearer token, which is valid for 1 month.
3. Provide your bearer token for subsequent API calls by clicking Authorize below or including in the header of all direct API calls: `Authorization: Bearer [your-token]`


Notes

1. "score" is a field across almost all entities. We generate scores for different types of entities in various ways, so no assumptions should be made about the meaning of this value. It is simply used to hint at relative popularity for sorting purposes.

Servers

https://api4.thetvdb.com/v4

▼


Authorize 

Login

POST

/login

create an auth token. The token has one month validation length.



Ejemplo de API pública

Curl

```
curl -X 'POST' \
'https://api4.thetvdb.com/v4/login' \
-H 'accept: application/json' \
-H 'Content-Type: application/json' \
-d '{
  "apikey": "8afd88e1-0b70-4e32-a333-9e8cf66df47f",
  "pin": "string"
}'
```

Request URL

<https://api4.thetvdb.com/v4/login>

Server response

Code	Details
------	---------

200

Response body

```

{
  "status": "success",
  "data": {
    "token": "eyJhbGciOiJIUzU1NiIsInR5cCI6IkpXVCJ9.eyJhZ2UiOiIiLCJhcGlrZXkiOiIiYWZkODhIMS0wYjcwLTRlMzIiYmZmY05ZTjZjY2ZGY0N2YiLCJjb2t1dW5pdHlf3c3VvcG9yZGVkIjpmYXZzZW5iZXhwIjoxNDM4MTM3ODg1L3NjZW5kZXIiOiIiLCJ0aXZz3B1c19kYXkiOiJlZWMDAwMDAwMwCwIAG10c19wXZjfbW9udGgiOiJlZWMDAwMDAwMwCwIawQ0iOiYnZyYmZyYiIiwiaXNfbn9kIjpmYXZzZW5iaXNf3c31zdGvXt2t1eSI6ZmFsc2UsImZlX3RydnXN0ZWQ0iOmzhbHhN1L3JwMjZlZDhjbmc1L3Jyb2xlcjIwM10sInR1bmFudC1InR2ZGIiLCJ1dWlkIjoiiIn0. XEjg7Toc6IFuiS0i5onxUcQj9jD_RfD4PUkugKSph0nw-GpUy168Ghg7gdq6U3Pyzf8B8sTmDBM64Pf_M_8V1fyh223hCgeQ2s0uwcDD01xs065mFJ0blxWXSpsKAZtXcm1Ax0L1_BHGrI5YeItea0EKOITZdrrwMgMaY5EKIVoZF8c1x17sjMm_XjpwGfprVXa21S09ISgLnJZ0GAlgmCue1E5J0y_em6r04121klYhLV0YUDnDZXQSP9b9aonCxaWRrKVpM0NXN_w9PMWODEFTZu3bbMYX68iCuxfYEPJWEkKcpgQj5uz5Mj4a51R3Q61_czPdeM_Uky6F_0cq0Be7071u1_d1w1z55Ma7mrKexY3GypxaFxnadQ1peuof9GmM_SMBN37b3fLnaYr4aYdMDDYGNTG6x5b30Rn0p8ZUKK8c-LdpD1q17697X2ceqEepPcDf9UjBnRaDxb57YzK_coGyXU0A51R80cx8gefHhcRu2U8YxIxtPrC8o2L3U2T2x4thWMRC77yFX_-W0MKirJX7pwZuXs5EVT59bpCnFtvBD8ae3sdFps5uQXdk1R2diptCcM00nKQMDuVtNPBdtVh0CR12_WsngB7VTR0s3jbbD3zBVDaxa01cqk3j8igOCj62CBEMD160MmVxv3i_7n3uHw0eTr7w2Q"
  }
}

```

Response headers

```
content-length: 901
content-type: application/json; charset=UTF-8
```

Responses

Code	Description
------	-------------

Links

200

No links

Ejemplo de API pública

Curl

```
curl -X 'GET' \
'https://api4.thetvdb.com/v4/movies/filter?country=usa&lang=eng&sort=score&year=2005' \
-H 'accept: application/json' \
-H 'Authorization: Bearer eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.eyJhZ2UiOiIiLCJhcG1rZXkiOiI4YWZkODhIMS0wYjcwLTRlMiItYTZmYy05ZThjZjY2ZGY0N2ViLCJjb21tdW5pdHlf1fc3VvcG9ydGVkIjpmYnZsZWwiZXhwIjoxNzM4MTM3ODg1LCJnZW50IjoiZm9udC8ifQ=='
```

Request URL

<https://api4.thetvdb.com/v4/movies/filter?country=usa&lang=eng&sort=score&year=2005>

Server response

Code

Details

200

Response body

```
{
  "status": "success",
  "data": [
    {
      "id": 564,
      "name": "Charlie and the Chocolate Factory",
      "slug": "charlie-and-the-chocolate-factory",
      "image": "/banners/movies/564/posters/564.jpg",
      "nameTranslations": null,
      "overviewTranslations": null,
      "aliases": null,
      "score": 3159110,
      "runtime": 115,
      "status": {
        "id": 5,
        "name": "Released",
        "recordType": "movie",
        "keepUpdated": true
      },
      "lastUpdated": "2024-12-26 20:00:29",
      "year": "2005"
    },
    {
      "id": 806,
      "name": "Mean Girls"
```

Fin del tema 7

Módulo de Desarrollo de Interfaces

2º DAM