

Proceso de desarrollo de API Twitch Analytics

Primeros pasos: Configuración de entorno y obtención de credenciales

El primer paso que se dio a la hora de comenzar con el proyecto era, evidentemente, encontrar la forma de obtener los datos de Twitch. Para ello nos registramos en Twitch Developers y generamos un Token inicial que guardamos en un documento de texto para hacer las pruebas locales.

Programación en local:

Creamos una aplicación de escritorio que funcionaba mediante un menú que permitía ejecutar los tres casos de uso, simplemente para verificar que conseguíamos conectarnos a la API de Twitch y obtener los datos que nos interesaban. Cuando este punto estaba completado decidimos pasar a la parte del servicio web.

Obtención y configuración del servidor web:

Obtuvimos una VPS “gratuita” a través de OracleCloud y guardamos las claves públicas y privadas que nos dieron. Después hicimos una conexión ssh con la clave privada para crear un usuario.

Para levantar el servidor accedimos a la web de Oracle y permitimos el tráfico entrante desde la 0.0.0.0 (cualquier dirección ip) por el puerto 80 y autorizamos el tráfico saliente por el 443. Desactivamos los firewalls para permitir conexiones.

Continuamos la configuración instalándole php, httpd (Apache 2), dependencias de json, curl...

Método de trabajo:

Inicialmente trabajamos de manera remota sobre la máquina del servidor a través de una conexión ssh, gestionando los archivos por terminal y editando los ficheros con nano. Después variamos entre utilizar FileZilla y conexiones ssh en phpStorm para la edición de archivos mediante el IDE.

Tras separar los distintos casos de uso en archivos diferentes, dimos los últimos retoques al funcionamiento de la web, como analizar las diferentes respuestas de los códigos de estado http y el quitar las extensiones .php de la URL que se pide utilizando un .htaccess.

Una vez todo esto estaba terminado hicimos comprobaciones (tanto manuales como usando PostMan) de los distintos endpoints. Al ver que el funcionamiento era correcto cerramos el proyecto y subimos todo a GitHub.

Problemas y deuda técnica:

- Control de versiones/GitHub: Gran parte del desarrollo se dió en el servidor de producción mediante conexiones ssh en vez de trabajar en los repositorios locales, y la subida de archivos al servidor se hacía mediante FileZilla o funcionalidades del IDE. Esta forma de trabajar no permite sacar provecho del mismo git y dificulta el reparto de tareas y el trabajo en paralelo entre otras cosas.
- Pruebas y desarrollo en producción: Consecuencia de lo anterior, una vez pudimos comprobar que conectábamos con la API de Twitch el resto del desarrollo y pruebas fue puramente en producción, por desconocimiento de métodos para trabajar en local en ese punto.
- Credenciales “hard-codeadas”: Al principio incluso el token estaba escrito directamente en los programas, eso lo arreglamos con la función desarrollada en token.php. Las credenciales de Twitch siguen estando hard-codeadas en ese código.
- Base de Datos: dado que solo había que implementar GETs y pudimos resolver el filtrado en el Caso de uso 3, nuestra aplicación hace una comunicación y traspaso de información directo con la API de Twitch, pero como idea a futuro será casi obligatorio introducir una BBDD que nos ayude tanto a filtrar como almacenar información.