Cómo Crear un Bot Telegram para ChatGPT

Este tutorial explica cómo crear un bot de voz para Telegram que integra Whisper y ChatGPT.

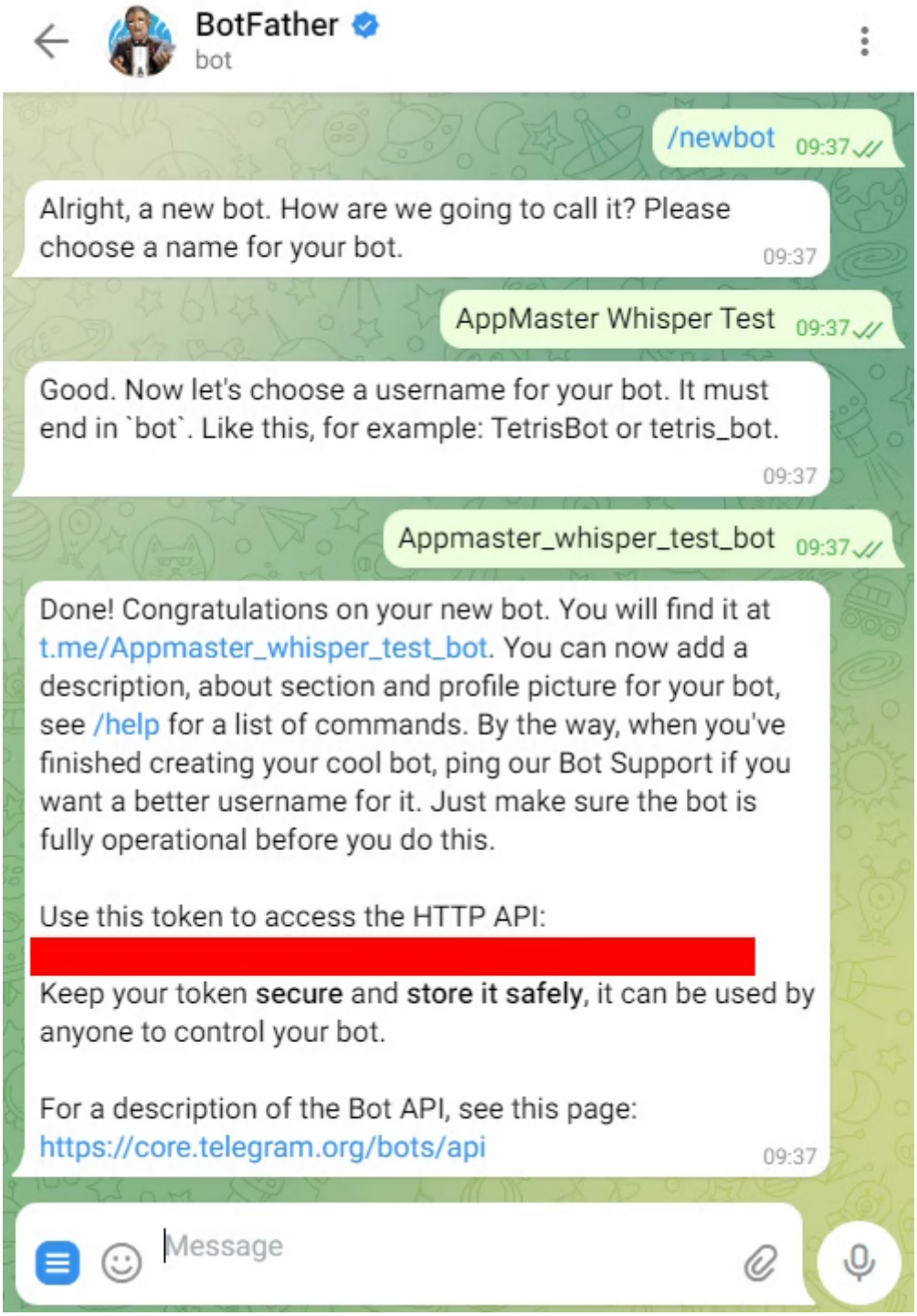
¡Todo NOCODE!, Hay un plan gratis para 2 apps web y 1 app móvil

Pasos para la Creación del Bot

1. **Creación de un Bot para Telegram y su Configuración Básica**
   - Registrar un bot en Telegram usando BotFather.
   - Obtener un enlace al bot y un token para las peticiones a su API.
2. **Recepción de un Archivo con un Mensaje de Voz**
   - Usar el módulo Telegram, diseñado para mensajes de texto, para recibir archivos de voz.
   - Analizar la estructura del mensaje y obtener parámetros como Sender ID y File ID.
3. **Convertir Archivo de OGA a MP3**
   - Utilizar servicios como Zamzar para convertir archivos OGA (formato de voz de Telegram) a MP3, compatible con Whisper.
4. **Envío de un Fichero MP3 a Whisper para su Conversión a Texto**
   - Preparar una solicitud en formato Multipart Form para enviar el archivo de voz a Whisper.
5. **Envío de un Texto Recibido a ChatGPT**
   - Utilizar HTTP Request o el módulo de AppMaster para OpenAI para enviar el texto a ChatGPT.
6. **Envío de una Respuesta de ChatGPT a un Usuario en Telegram**
   - Repetir el proceso de envío de mensajes a Telegram, pero esta vez con la respuesta de ChatGPT.

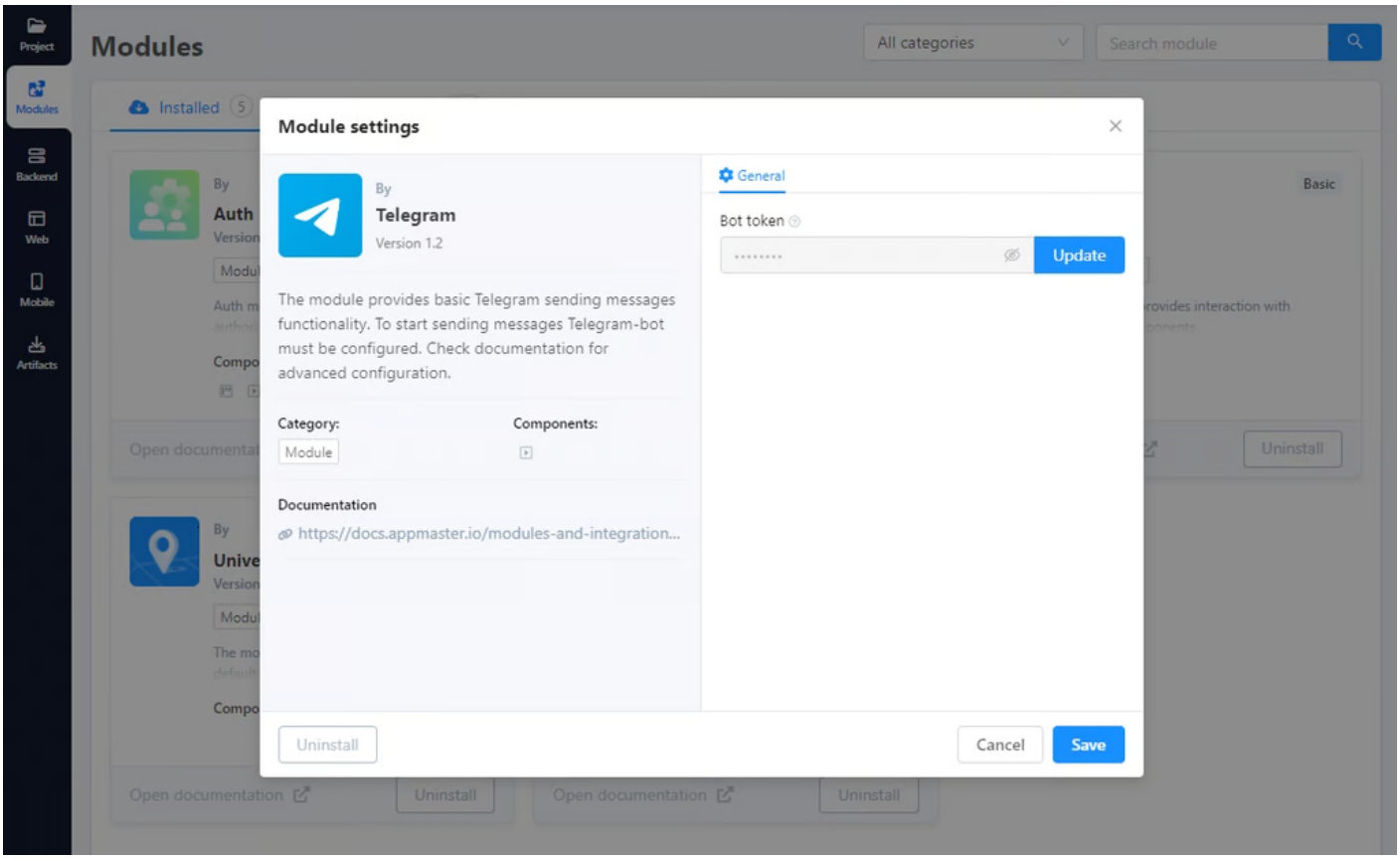
Creación de un bot para Telegram y su configuración básica

Todo comienza con el hecho mismo de dar de alta un bot. Y para ello, Telegram utiliza su propio bot especial - . BotFather. Tienes que escribirle y seguir unos sencillos pasos secuenciales.

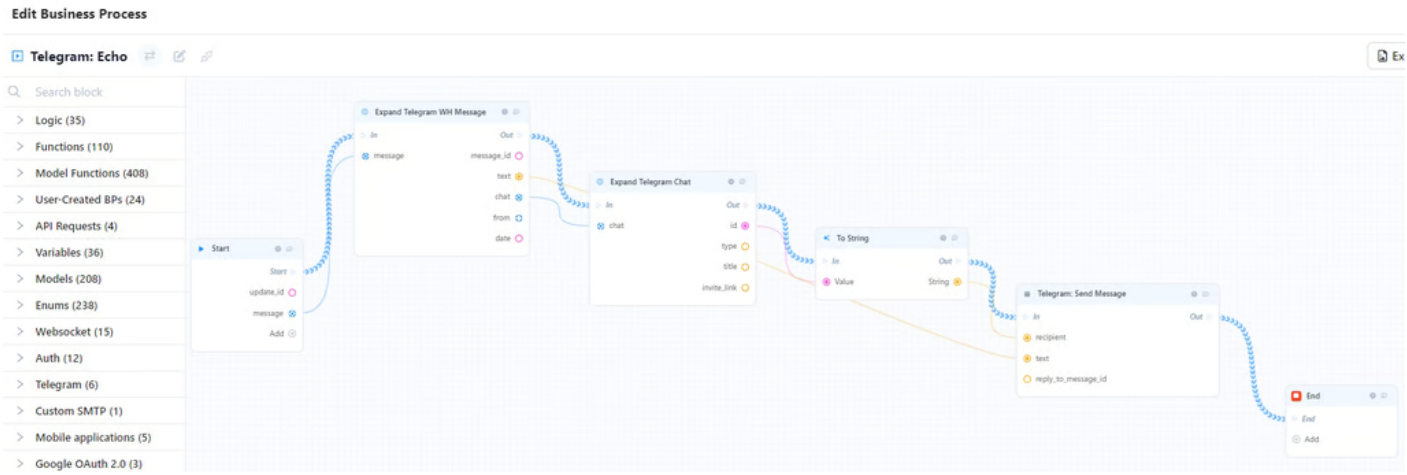


Como resultado, deberías tener un enlace a tu bot y un token para las peticiones a su API.

Ahora puedes pasar a configurar el bot en AppMaster. La forma más sencilla de hacerlo es instalar el módulo apropiado. Sólo tienes que seleccionarlo en la lista de módulos y especificar tu token de API en la configuración.

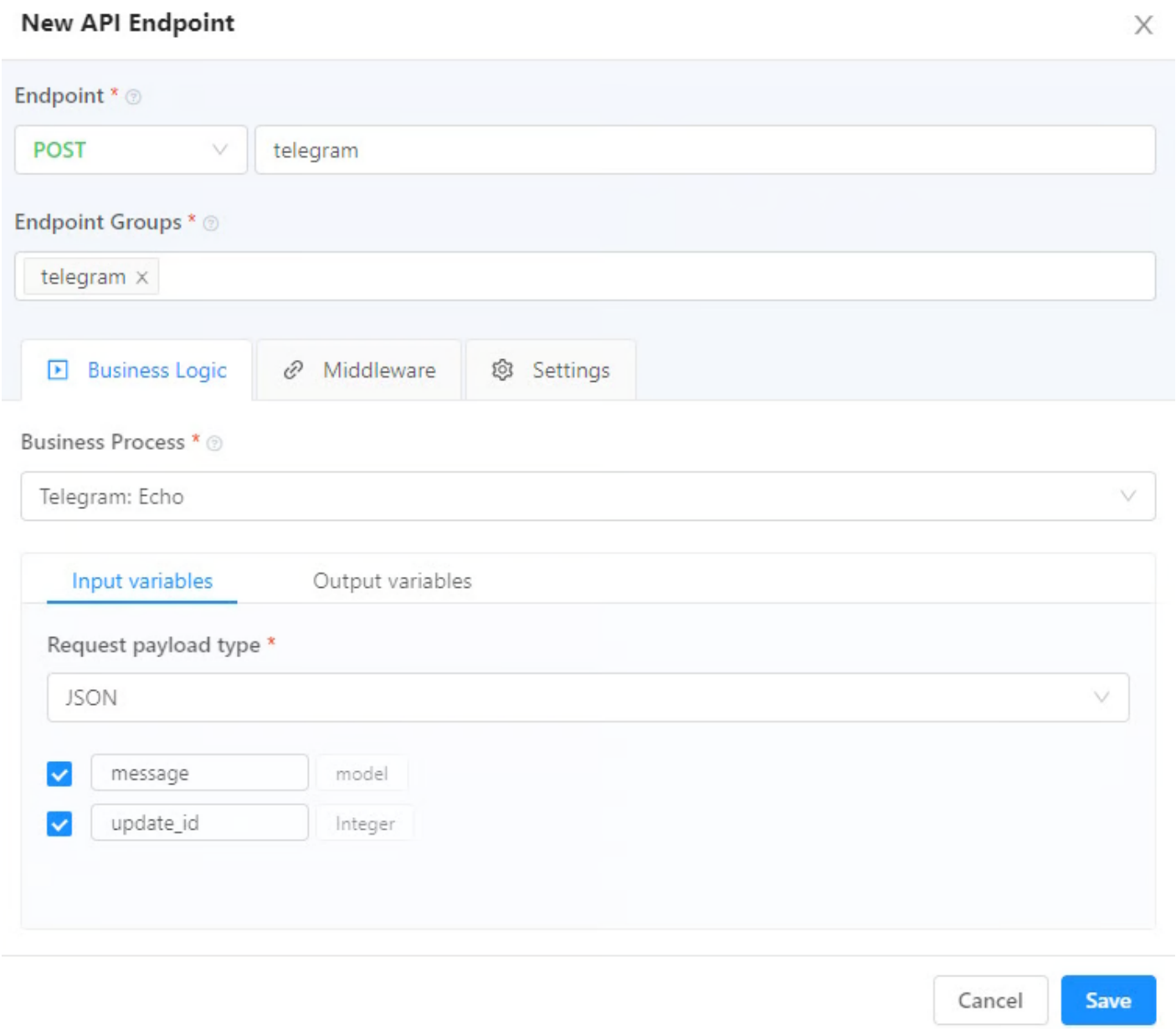


Los modelos necesarios se crearán automáticamente en tu proyecto, aparecerán bloques para trabajar con Telegram, así como el proceso de negocio básico de Telegram Telegram: Echo. Vale la pena considerar cuidadosamente y entender el principio de trabajo.



En la entrada, recibe un mensaje de Telegram (modeloMessage). Contiene el texto del mensaje enviado (text), así como el modelo chat del que puede obtener el identificador del remitente (id). El último bloque Telegram: Send Message devuelve el mensaje recibido al remitente, pero en nombre del bot.

Puedes utilizar este proceso de negocio para las pruebas iniciales del bot y la comunicación con él. Para ello, necesitas crear un endpoint que reciba información e inicie un proceso de negocio.



Al crearlo, es importante desactivar Middleware Token Auth. El endpoint debe estar abierto para su uso sin autorización.

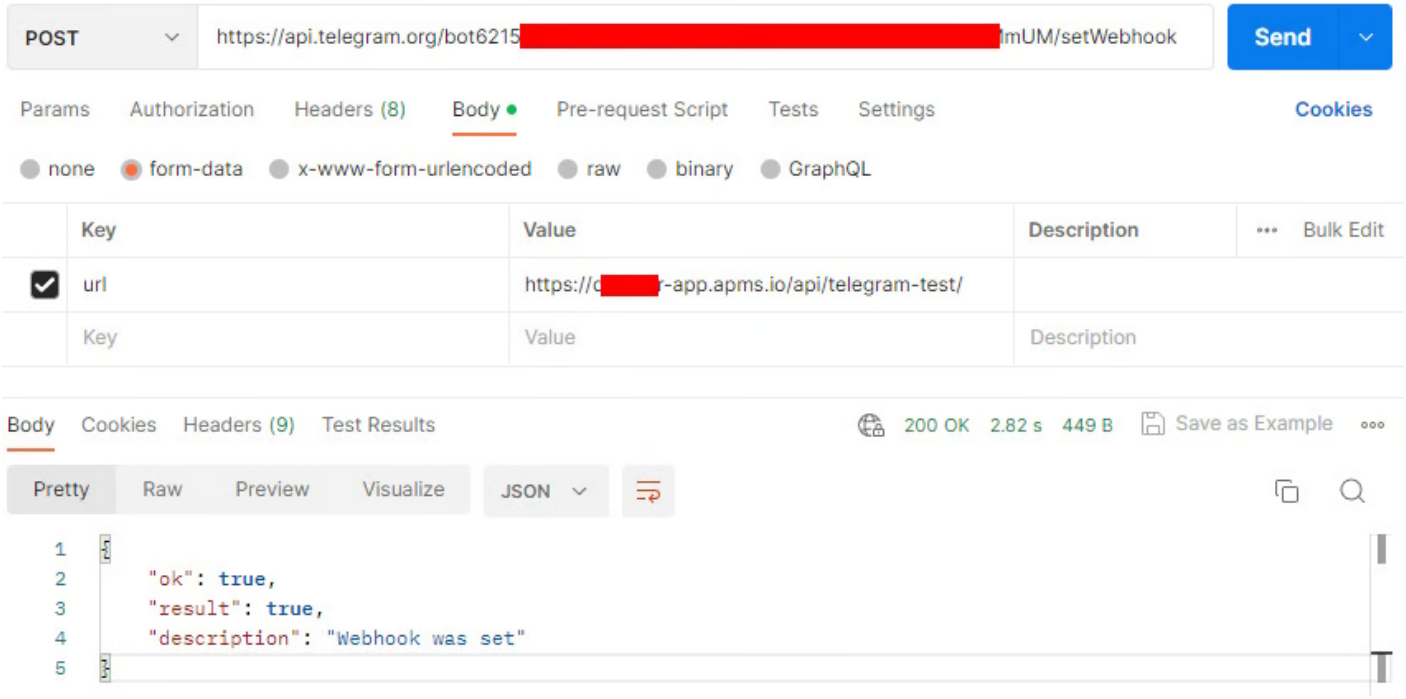
El principio de funcionamiento del bot Telegram es bastante simple - todos los mensajes que le serán enviados van a un webhook especial, que puede transferirlos automáticamente más allá y enviarlos al endpoint de su elección para acciones posteriores.

En consecuencia, el último paso que queda es activar el bot - es necesario registrar este punto final en Telegram e indicar que este bot debe estar asociado con él. Para ello, debe enviar una solicitud a POST con la URL completa del endpoint e indicar su token de bot en lugar de {Bot API Token} a la dirección

https://api.telegram.org/bot{Bot API Token}/setWebhook

Si recibes un mensaje como respuesta, entonces todo se ha hecho correctamente.

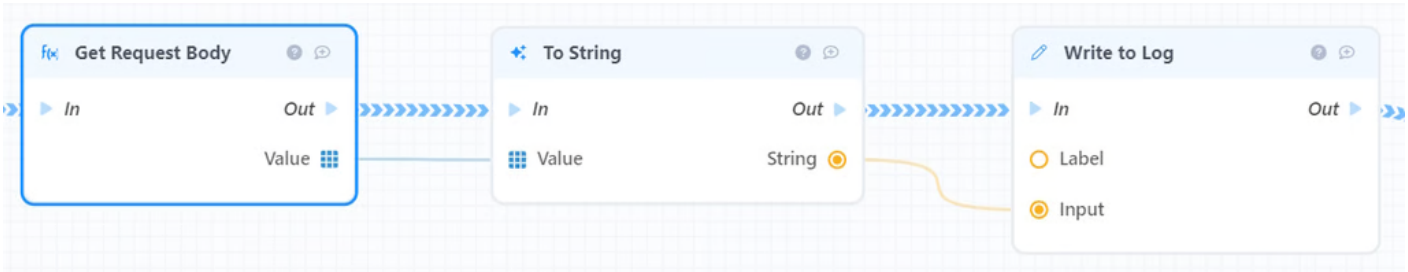
```
{
 "ok": true,
 "result": true,
 "description": "Webhook was set"
}
```

El bot está listo para trabajar, puedes enviarle un mensaje y recibirlo de vuelta.

Recibir un archivo con un mensaje de voz

El módulo Telegram está diseñado principalmente para trabajar con mensajes de texto. Y nuestra tarea es recibir un archivo con un mensaje de voz. Con AppMaster, puedes resolver este problema fácilmente. Primero, necesitamos analizar lo que generalmente recibimos de Telegram para analizar la estructura del mensaje. El bloque Get Request Body está diseñado para ello. Elimina la necesidad de especificar la estructura de la solicitud de antemano y permite recibir la solicitud completa, independientemente de su contenido. Este bloque devuelve el resultado de la consulta como un conjunto de bytes, y puede utilizar el bloque To String para presentar el resultado de forma legible, así como para guardarlo en registros (Write to Log) para su posterior análisis.



Nos interesan dos parámetros de la petición completa:

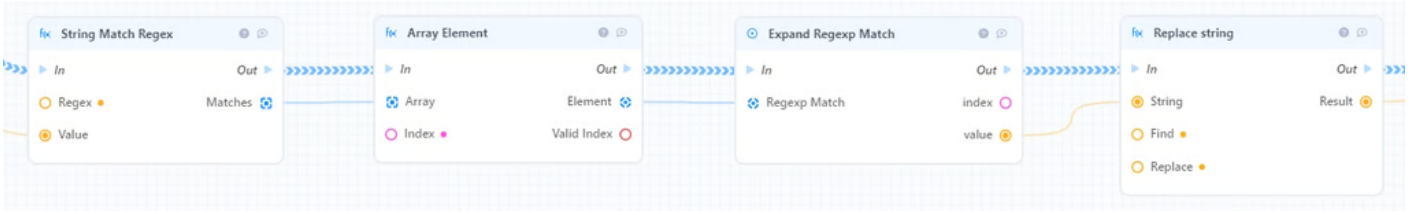
Sender ID - especificados en la petición como "id":300493858 File ID - "file_id":"AwACAgIAAxkBAAMzZBk6QRvO-OYWsWUC-Bu1UXDM2FwAAkktAAKTZclIWTSkfTTw8wYvBA"

Puede crear su propio modelo que coincida con la solicitud y utilizarlo para obtener los campos requeridos. Pero será más rápido crear una expresión regular (Regex) y utilizarla. Para ello, el bloque String Match Regex toma la propia expresión como entrada, así como la cadena en la que se comprobará la coincidencia con la expresión dada.

En el primer caso, la expresión es "id":"\d+

Como resultado, obtenemos la cadena "id":300493858, de la que tendremos que eliminar el extra ("id":) utilizando el bloque Replace String y dejar sólo el propio identificador.

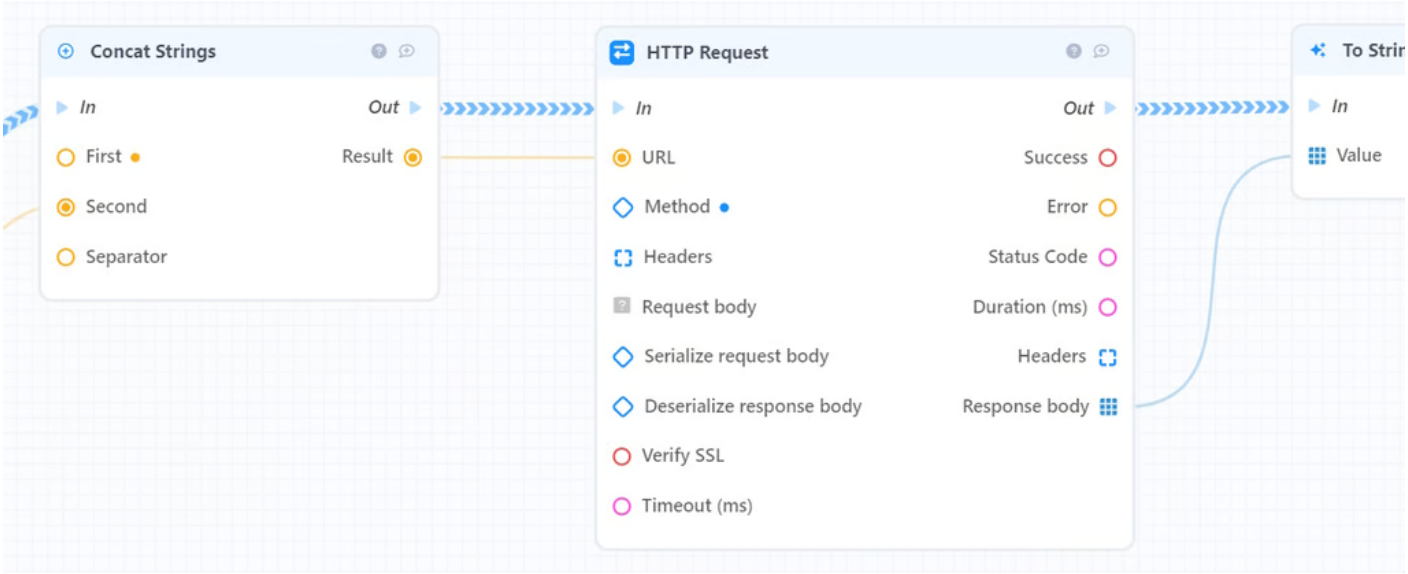
En el segundo caso, el principio es exactamente el mismo, pero se utiliza una expresión ligeramente más compleja: "file_id":"[^"]+



Ahora tenemos el identificador del remitente y el identificador del archivo, y podemos usarlo para obtener el archivo en sí. Para ello, hay que dirigirse a Telegram API. Esto ya se ha hecho antes al registrar el endpoint del bot. Ahora necesitas hacer una petición similar para obtener el archivo. {File ID} en la URL de la petición debe ser reemplazado por el ID del archivo recibido.

https://api.telegram.org/bot{Bot API Token}/setWebhook/getFile?file_id={File ID}

Para enviar una petición y recibir su resultado, utilizamos el bloque HTTP Request especificando URL y Method = GET como parámetros.



A partir de la respuesta recibida, se puede averiguar la ruta relativa al archivo, se pasa en el parámetro "file_path". En consecuencia, utilizando la siguiente expresión regular ("file_path":"[^"]+) puede extraer el valor deseado y conectar con "https://api.telegram.org/file/bot{Bot API Token}/" para obtener el enlace completo al archivo.

Convertir archivo de OGA a MP3

Se recibe el archivo, pero el obstáculo es que el Whisper no permite trabajar con el formato OGA formato. Es necesario convertirlo a uno de los formatos adecuados.

Como ejemplo, se utiliza el servicio Zamzar (su plan gratuito permite realizar 100 conversiones al mes) y convertir a MP3.

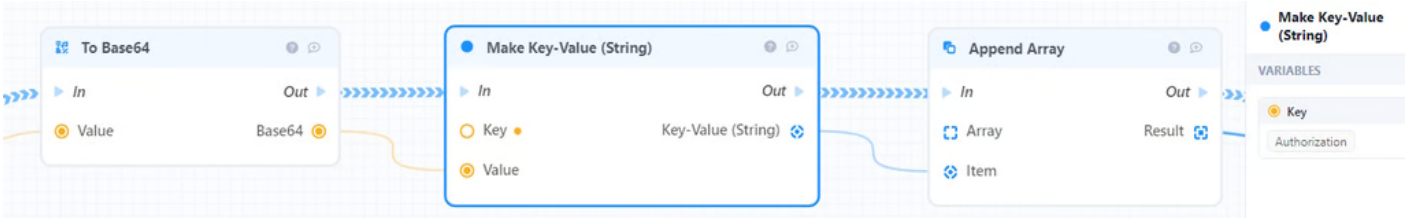
Puede consultar su documentación para más detalles o utilizar otro servicio similar. No analizaremos el trabajo con él en detalle, y sólo consideraremos la parte que se relaciona directamente con la implementación de AppMaster.

En primer lugar, la solicitud necesitará los datos de autenticación correctos. Deben proporcionarse en el formato Basic Authentication . Para ello, es necesario pasar una cabecera con valores en la solicitud:

Key = 'Authorization'

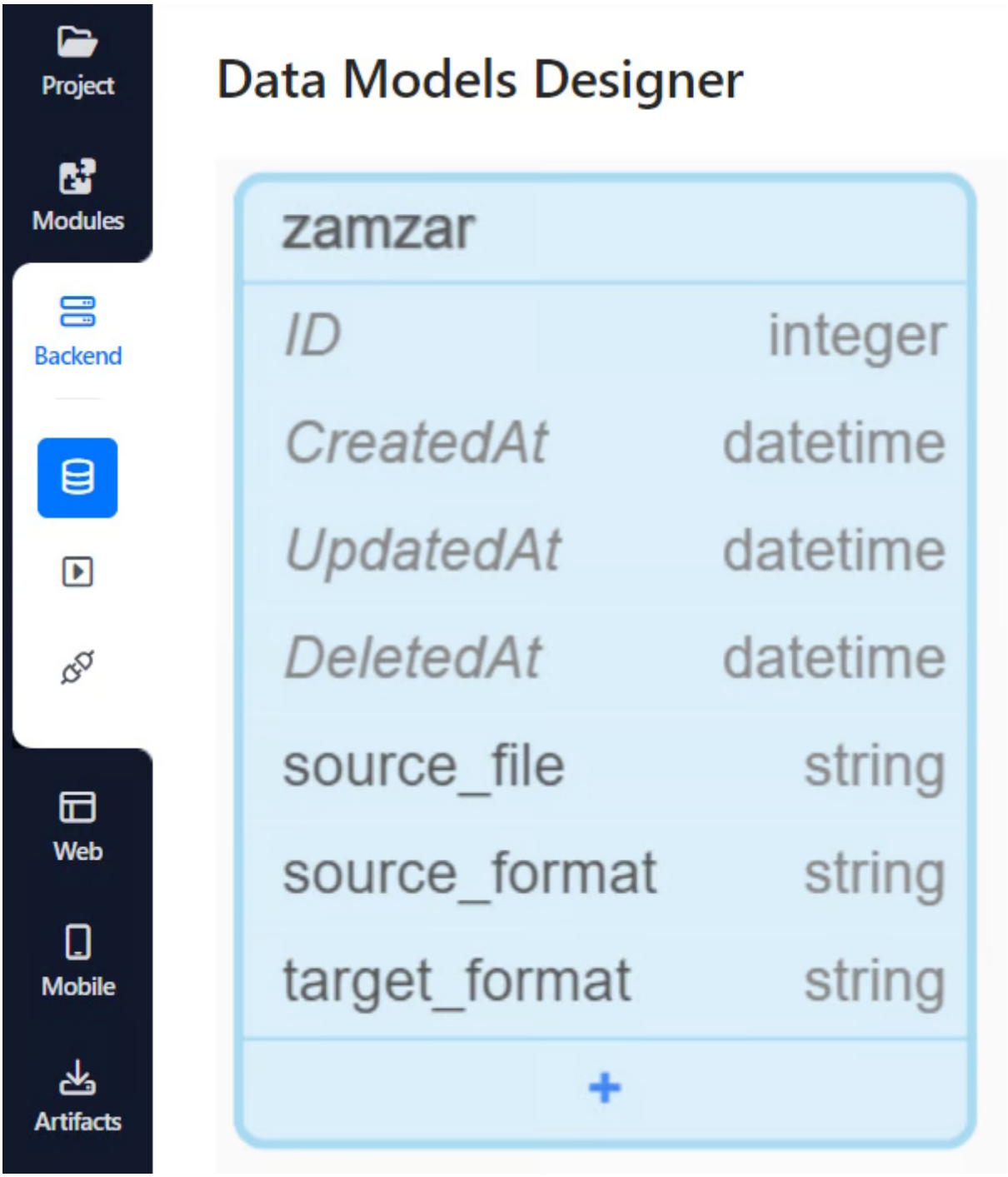
Value = 'Basic '+ ID de usuario y contraseña separados por ":" en base64 formato

La clave API obtenida al registrarse en el servicio es el ID de usuario. Es necesario añadirle ":" y codificarla en formato Base64 utilizando el bloque To Base64 bloque. El resultado debe convertirse en una cabecera (Make Key-Value (String) bloque).

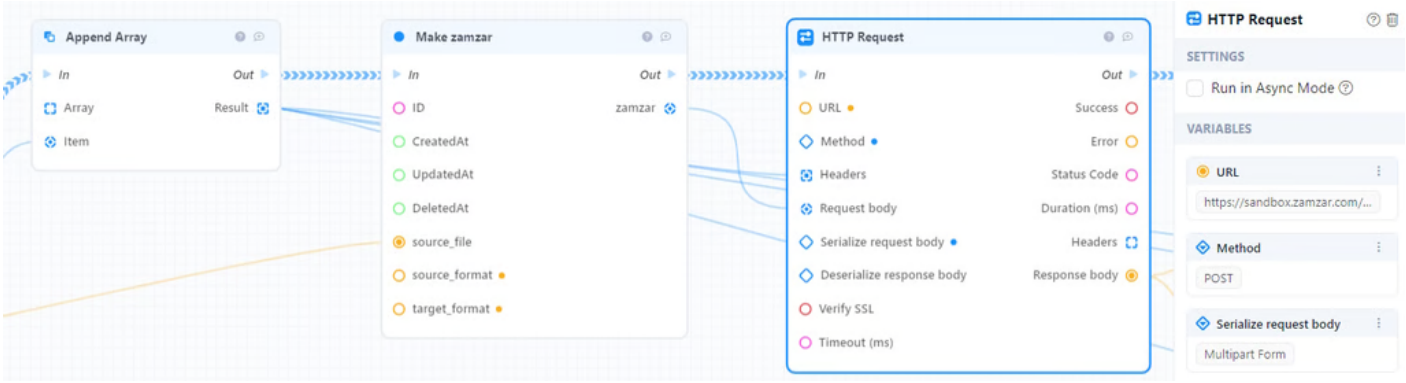


El siguiente paso es crear un modelo para la consulta en el diseñador de bases de datos. La consulta debe enviarse en el formato Multipart Form respectivamente, es necesario preparar un modelo de la forma de esta petición. En nuestro ejemplo, el modelo consta de tres campos de tipo String:

- source_file - la ruta completa al archivo de origen (se aprendió en el paso anterior).
- source_format - formato del archivo de origen, en este ejemplo, es un valor fijo "ogg".
- target_format - formato de destino para la conversión. Puede elegir cualquier formato soportado por Whisper. Utilicemos "mp3" como ejemplo.

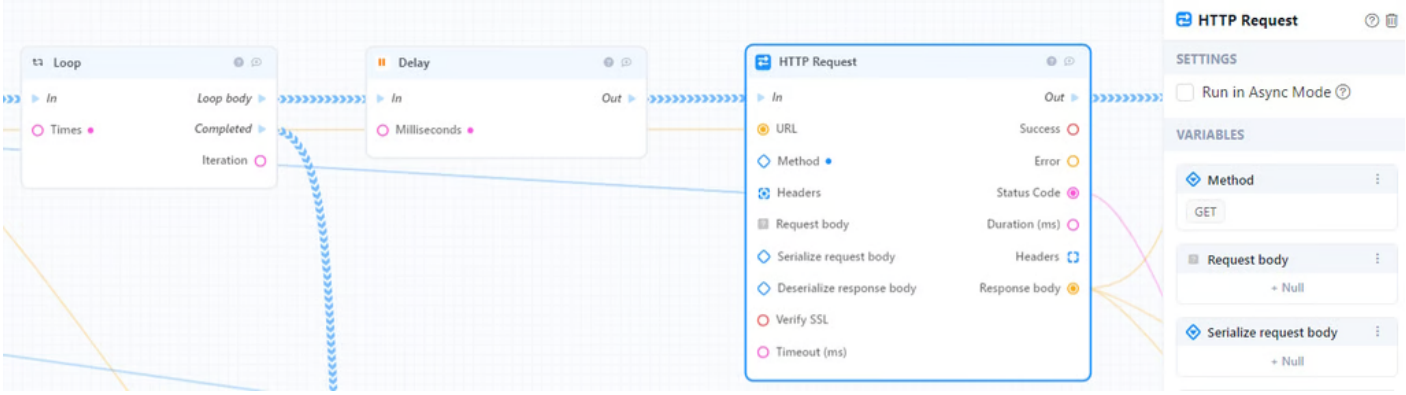


En el editor de procesos de negocio, debe utilizar el bloque Make para rellenar los datos del modelo y enviarlos como una solicitud POST a https://sandbox.zamzar.com/v1/jobs/ utilizando el bloque HTTP Request (asegúrese de especificar Serialize request body = Multipart Form).



Hay que tener en cuenta que esta solicitud no devuelve el archivo convertido, sino que sólo crea una tarea para convertirlo. Es necesario solicitar el resultado por separado; para ello, se necesita el ID de la tarea creada. Este ID debe obtenerse del cuerpo de la respuesta a la solicitud, y para ello debe realizarse el proceso ya elaborado utilizando expresiones regulares y extrayendo el valor del id.

El resultado de la conversión debe aplicarse por separado. Esto requerirá dos peticiones más. La primera es averiguar si el resultado está listo. La segunda es para recoger el fichero terminado. Al mismo tiempo, no sabemos el tiempo exacto de preparación, por lo que podemos organizar un bucle que envíe peticiones repetidas para comprobar la preparación a determinados intervalos (por ejemplo, cada segundo).



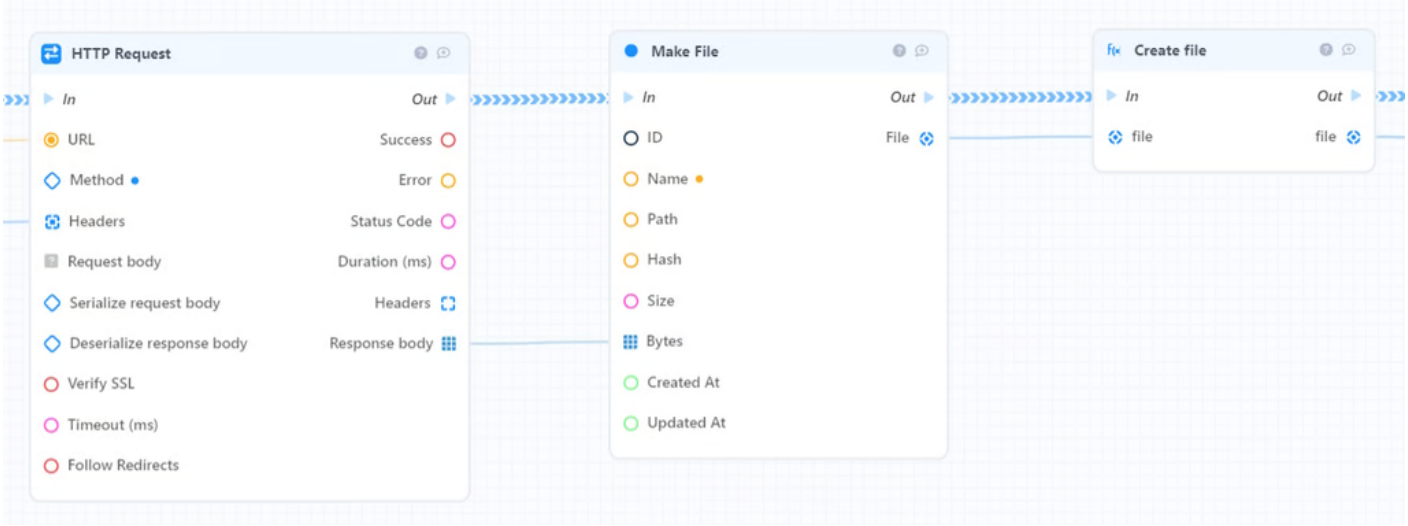
Se debe enviar un HTTP Request utilizando el método GET a la URL https://sandbox.zamzar.com/v1/jobs/{id}, donde {id} es el id de tarea obtenido en el paso anterior. Para ello se utilizan las mismas cabeceras que en la petición anterior.

A partir de la respuesta recibida, hay que averiguar el estado de preparación. Si la conversión se ha completado, la respuesta contendrá "status": "successful" y para nosotros, esto es una señal de que podemos completar el bucle y seguir adelante.

Además del estado, la respuesta debe contener el ID del archivo terminado ("target_files":[{"id":). Debe extraerse para obtener el enlace final al archivo de la forma https://sandbox.zamzar.com/v1/files/{ID}/content

Al mismo tiempo, la recepción de un archivo sólo está disponible para los usuarios autorizados, por lo que es necesario ejecutar un HTTP Request utilizando las mismas cabeceras que en las solicitudes anteriores.

Como resultado de la solicitud, se obtendrá el contenido del archivo, al que se debe dar un nombre y guardarlo para su uso posterior.

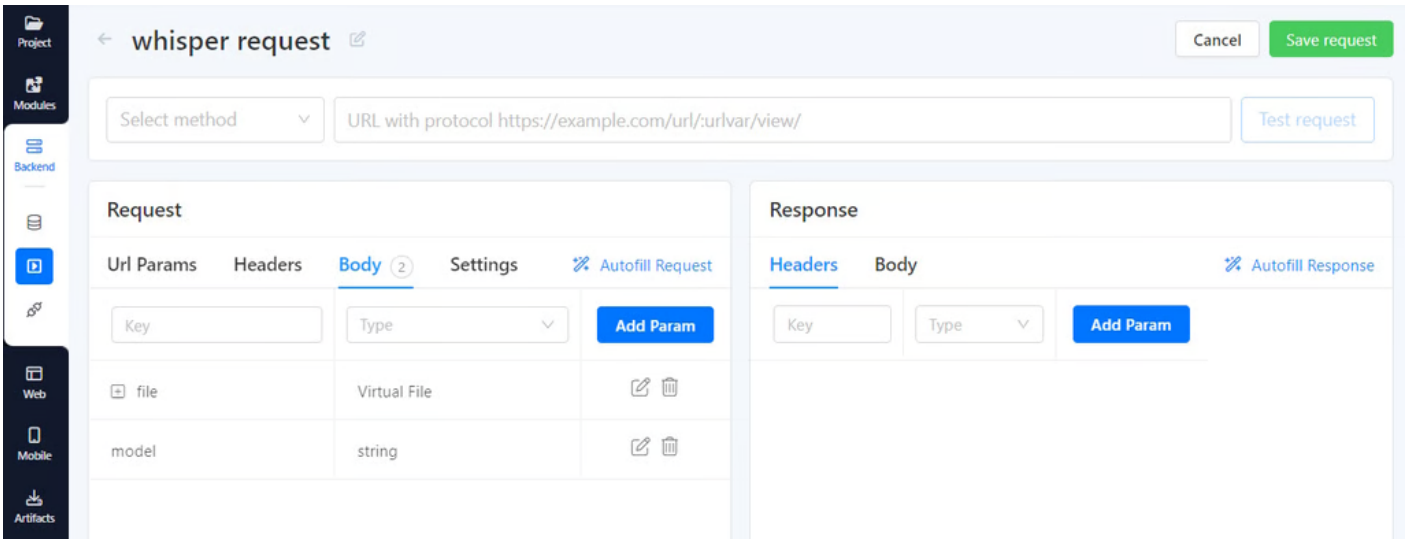


Envío de un fichero MP3 a Whisper para su conversión a texto

Ahora todo está listo para el siguiente paso: enviar un archivo con un mensaje de voz para su reconocimiento. Esto requerirá otra solicitud en el formato Multipart Form formato. Sólo que, a diferencia del ejemplo anterior, la solicitud tendrá que transferir el propio archivo y no un enlace al mismo.

Se puede crear un modelo para dicha solicitud en la sección External HTTP Request sección En este caso, no puede crear una solicitud por completo, sino limitarse a crear un modelo de cuerpo de solicitud. El modelo propiamente dicho consta de dos parámetros

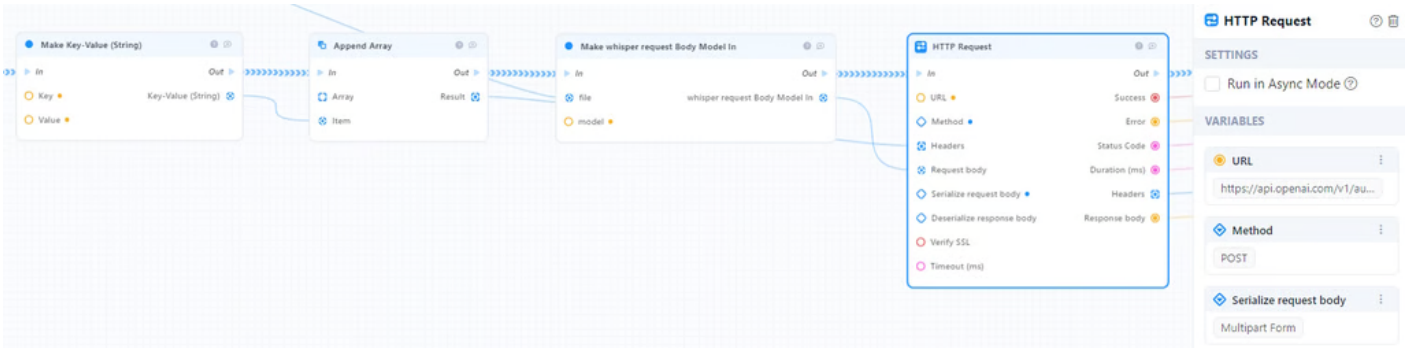
- File (Virtual File type) - el mismo fichero que hay que reconocer.
- model (tipo String) - aquí especificamos el valor whisper-1.



Además, para la solicitud, es necesario obtener una clave para trabajar con la OpenAI API y generar una cabecera de autorización del Bearer Token tipo .

- Key = 'Authorization'
- Value = 'Bearer '+ OpenAI API Key

A continuación, puede enviar la propia solicitud POST para reconocer el mensaje de voz al servicio Whisper en la URL https://api.openai.com/v1/audio/transcriptions

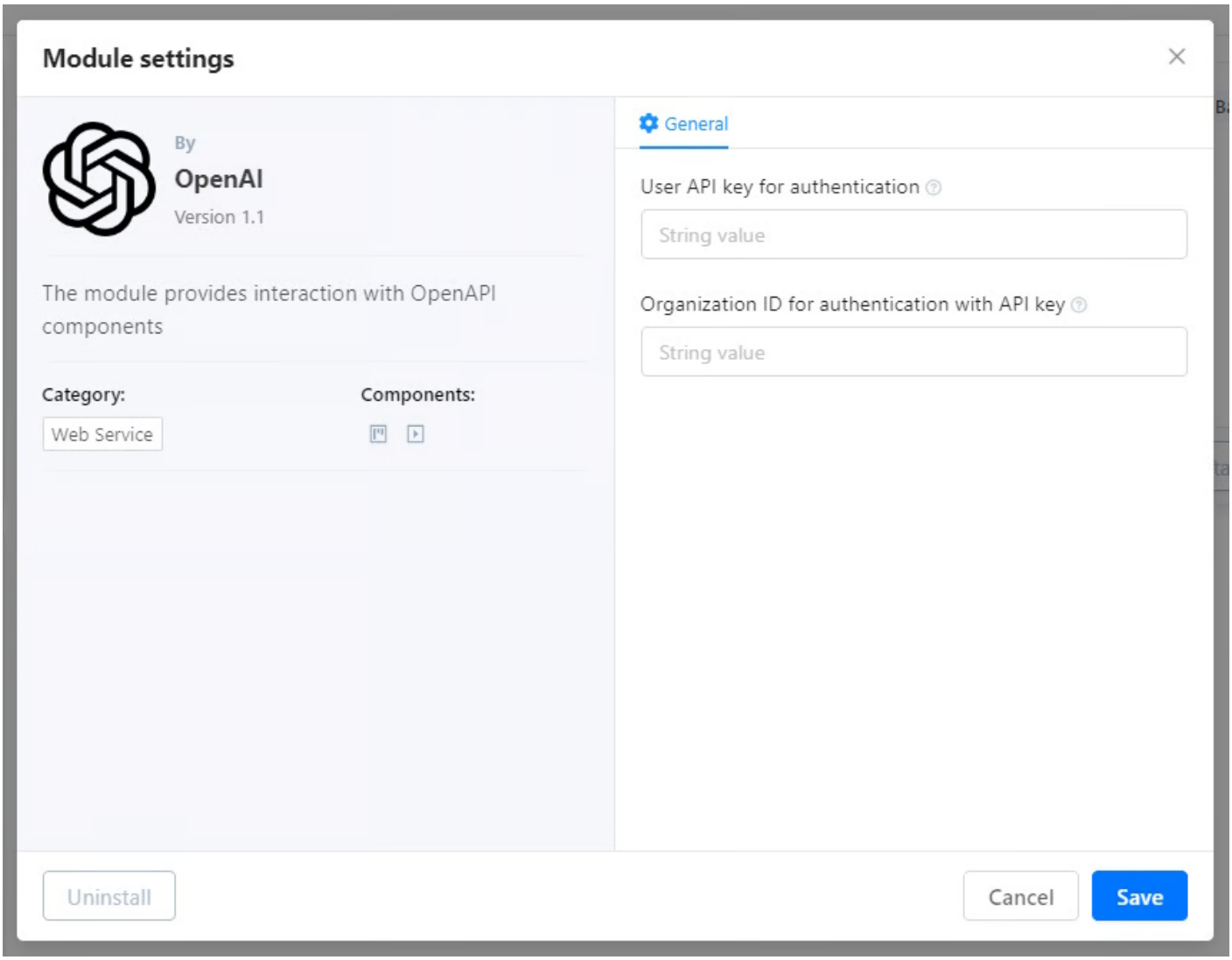


Como resultado del reconocimiento satisfactorio del archivo, se recibirá una respuesta del tipo

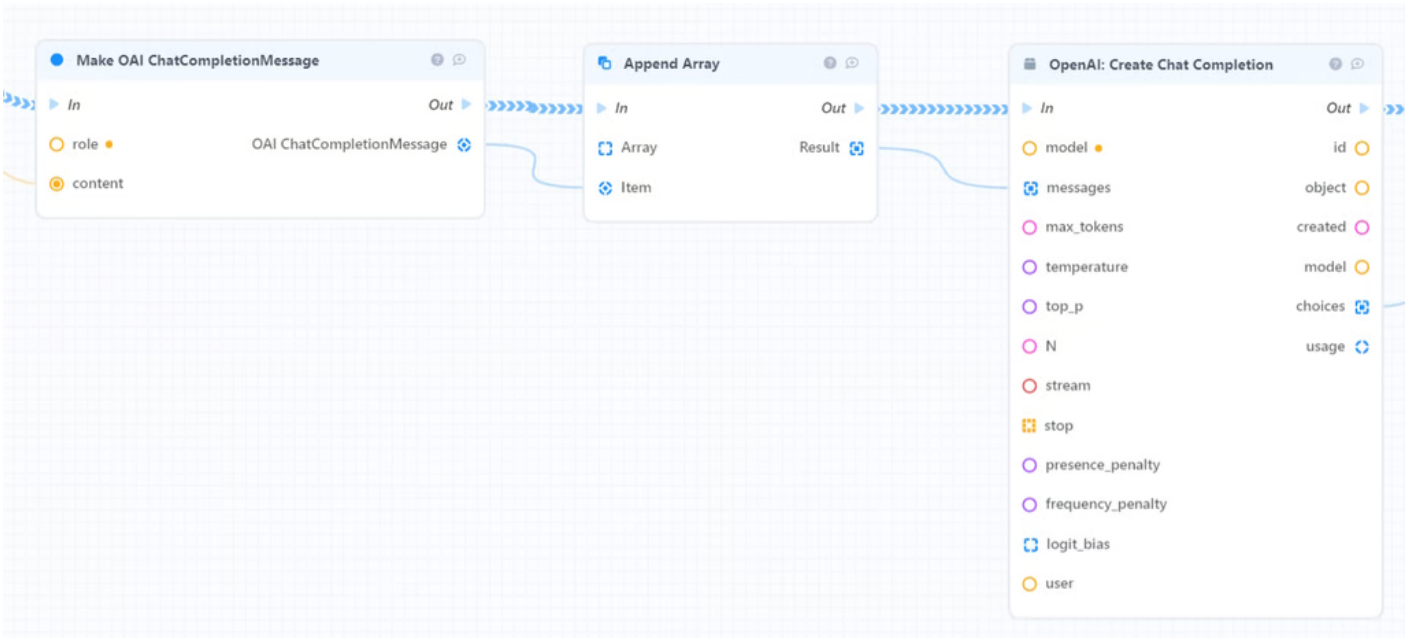
```
{"text": "Hello world.”}
```

Envío de un texto recibido a ChatGPT

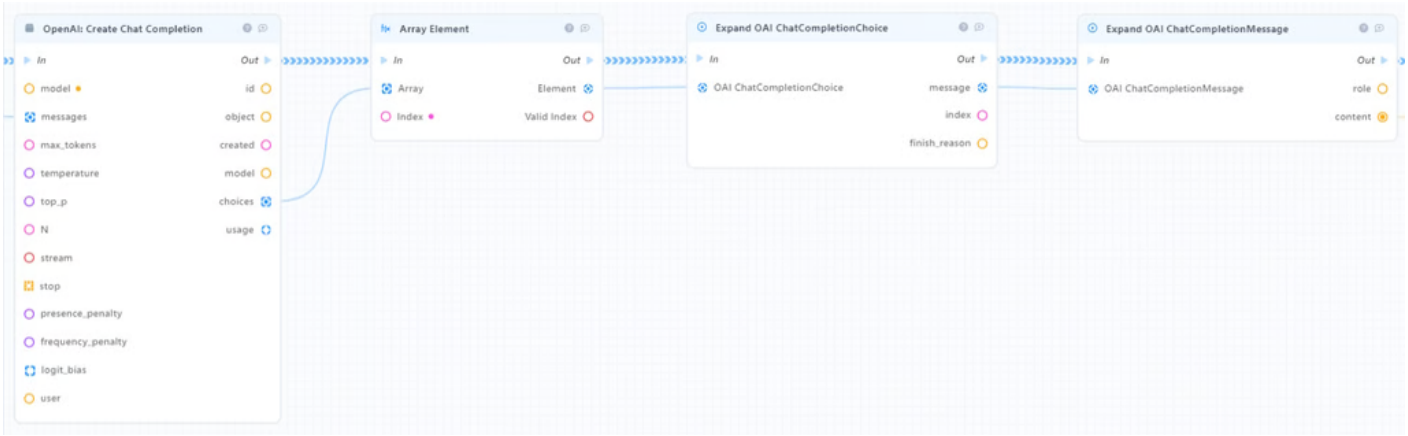
Puede seguir utilizando HTTP Request para enviar una solicitud a ChatGPT. para explorar la documentación de la API, así como para crear de forma independiente modelos de peticiones y respuestas. Pero también puede utilizar una opción más sencilla en forma de módulo ya preparado de AppMaster para trabajar con OpenAIque debe instalarse en la sección de módulos.



En la versión mínima suficiente, sólo es necesario especificar los parámetros del OAI ChatCompletionMessage (role = user, content = mensaje a enviar), añadirlo a la matriz, y enviar una petición a ChatGPT con el bloque OpenAI: Create Chat Completion (establecer parámetro model = gpt-4).

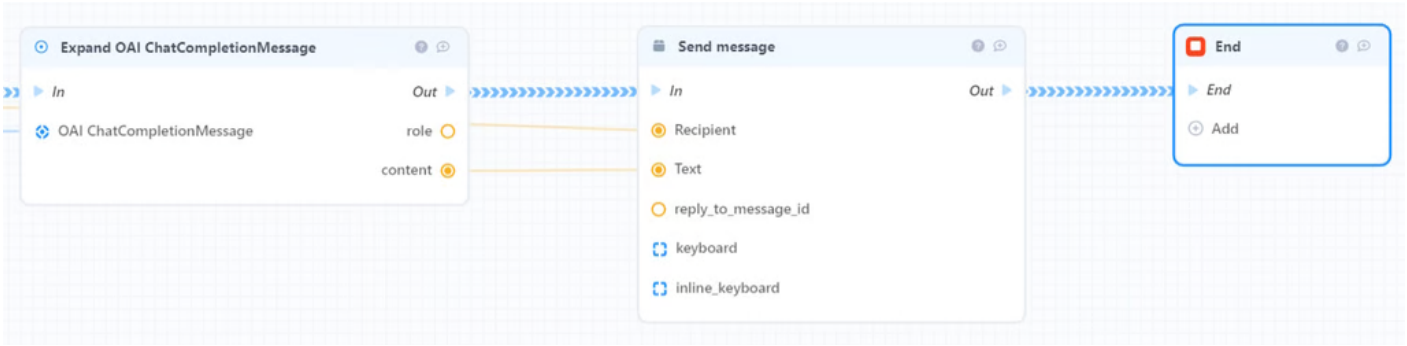


Como resultado, obtenemos una respuesta de ChatGPT. La leemos del content del modelo OAI ChatCompletionChoice modelo.



Envío de una respuesta de ChatGPT a un usuario en Telegram

El último paso es sólo repetir lo que ya se ha hecho antes - enviar un mensaje a Telegram. Pero si empezamos simplemente devolviendo el mensaje al remitente, ahora este mensaje ha puesto en marcha una serie de diversas acciones y el resultado se devuelve como una respuesta de ChatGPT.



En el proceso de desarrollo de un bot de este tipo, merece la pena considerar:

1. Telegram trabaja de tal manera que cada solicitud enviada por el bot debe ser procesada con éxito. De lo contrario, intentará repetirla muchas veces, lo que significa que en caso de problemas la recibirás una y otra vez. Asegúrese de que la lógica está construida de tal manera que la solicitud no dé lugar a un error y pueda ser procesada con éxito.
2. No todas las solicitudes funcionarán según lo previsto en el primer intento. Tendrá que buscar errores, y para ello disponga de Write to Log bloques. Son útiles tanto para comprobar que cada bloque recibe realmente los datos que tenía previsto transmitir como para analizar el resultado del bloque y estudiar la respuesta en detalle.
3. Como parte de la guía, hemos supuesto que todas las peticiones son predecibles y tienen éxito y que el proceso es bastante lineal. En la práctica, será necesario proporcionar una variedad de opciones para las acciones (por ejemplo, qué hacer si el usuario envía un mensaje de texto en lugar de un mensaje de voz o incluso una imagen) y la gestión de errores (comprobar el estado de las solicitudes HTTP, tener en cuenta diferentes opciones

de respuesta).

Consideraciones Importantes

- Asegurar que cada solicitud del bot sea procesada con éxito en Telegram.
- Realizar pruebas exhaustivas y manejar posibles errores.
- Prever variabilidad en las acciones y respuestas del bot.

Este tutorial proporciona una guía detallada para integrar eficazmente Telegram, Whisper y ChatGPT en un bot funcional.

Source

Revisión #4
Creado 20 noviembre 2023 10:56:58 por Tom
Actualizado 20 noviembre 2023 11:41:35 por Tom