**Telegram Bot**

*Cloud Functions*

Ссылка для вызова:

<https://functions.yandexcloud.net/d4erdi2u7dd75398752e>

import json

import requests

import telebot

import os

API\_TOKEN = os.environ.get('TELEGRAM\_BOT\_TOKEN')

if not API\_TOKEN:

    raise ValueError("Telegram bot token not provided in the environment variables.")

bot = telebot.TeleBot(API\_TOKEN)

LIBRETRANSLATE\_URL = "https://translate.flossboxin.org.in/translate"

TARGET\_LANGUAGE = "en"

def translate\_text(text, target\_language):

    """Переводит текст на указанный язык с помощью LibreTranslate."""

    try:

        response = requests.post(

            LIBRETRANSLATE\_URL,

            json={

                "q": text,

                "source": "auto",  # Автоматическое определение языка

                "target": target\_language,

                "format": "text"  # Указываем, что формат текста - обычный текст

            }

        )

        response.raise\_for\_status()  # Поднимает исключение для HTTP ошибок

        return response.json()["translatedText"]

    except requests.exceptions.RequestException as e:

        return f"Ошибка перевода: {e}"

@bot.message\_handler(commands=['start', 'help'])

def handle\_start\_help(message):

    """Обрабатывает команды /start и /help."""

    if message.text == "/start":

        bot.reply\_to(message, "Привет! Я бот-переводчик. Просто отправь мне текст, и я переведу его на английский.")

    elif message.text == "/help":

        bot.reply\_to(message, "Я перевожу любой текст на английский язык. Используйте команду /start для начала работы.")

@bot.message\_handler(func=lambda message: True)

def translate\_message(message):

    """Переводит текст, отправленный пользователем."""

    translated\_text = translate\_text(message.text, TARGET\_LANGUAGE)

    bot.reply\_to(message, translated\_text)

def handler(event, context):

    """Обработчик Cloud Functions."""

    try:

        json\_string = event['body']

        update = telebot.types.Update.de\_json(json\_string)

        bot.process\_new\_updates([update])

        return {

            'statusCode': 200,

            'body': 'OK',

        }

    except Exception as e:

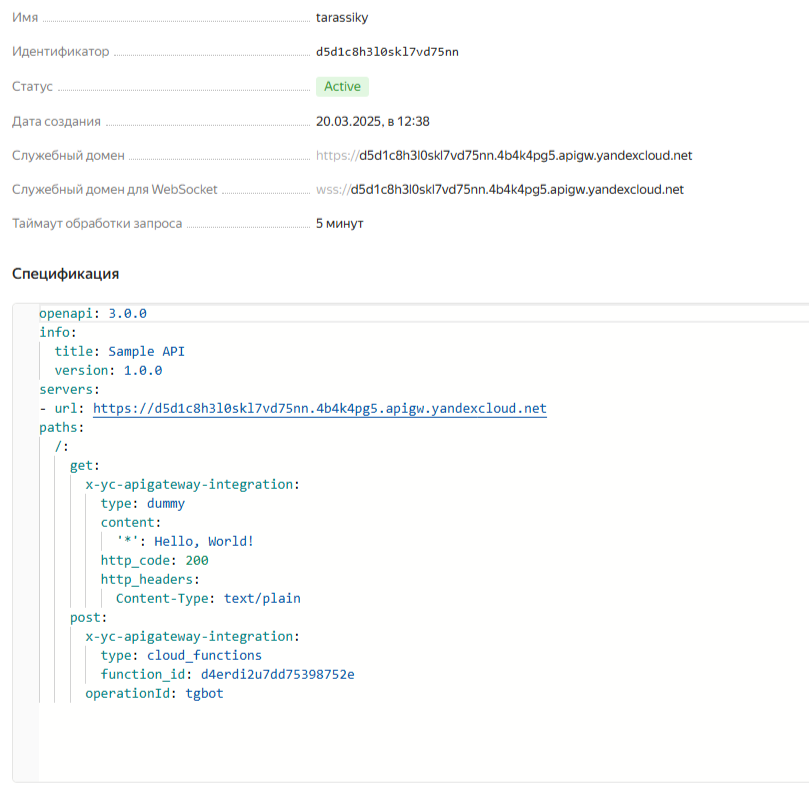
        return {

            'statusCode': 500,

            'body': f"Error processing request: {e}",

        }

# *API-шлюз*



*Ник*

@tarassiky\_bot