По совету Васи Комышева реализована связка ТГ с локальным n8n

Привет! 😊 Отличный вопрос — и очень полезный для будущей самостоятельной работы.

Если бы ты хотел **получить такой бот с нуля**, вот как мог бы выглядеть пример **грамотно составленного промпта**, чтобы получить в ответ тот код, который мы с тобой реализовали:

**🔹 Пример промпта:**

Хочу создать Telegram-бота на Python, который будет служить посредником между пользователем и моим локальным сервером n8n.

✅ Задача:

* Бот принимает сообщение от пользователя.
* Пересылает его на **локальный webhook в n8n**.
* После обработки в n8n, получает **ответ обратно через отдельный POST-запрос от n8n** и отправляет его в Телеграм.

⚙️ Условия:

* n8n работает локально, вебхуки имеют формат http://localhost:5678/webhook/....
* Нужно обойти ограничение Telegram (он не работает с http://localhost) — поэтому бот сам делает GET-запрос к локальному вебхуку.
* В ответ от n8n приходит JSON с полями chat\_id и reply.

💡 Желательно:

* Использовать aiogram, Flask и requests.
* Обработку сообщений через polling (не через webhook).
* Возможность хранить BOT\_TOKEN в .env файле.

Напиши, пожалуйста, весь код такого бота с комментариями.

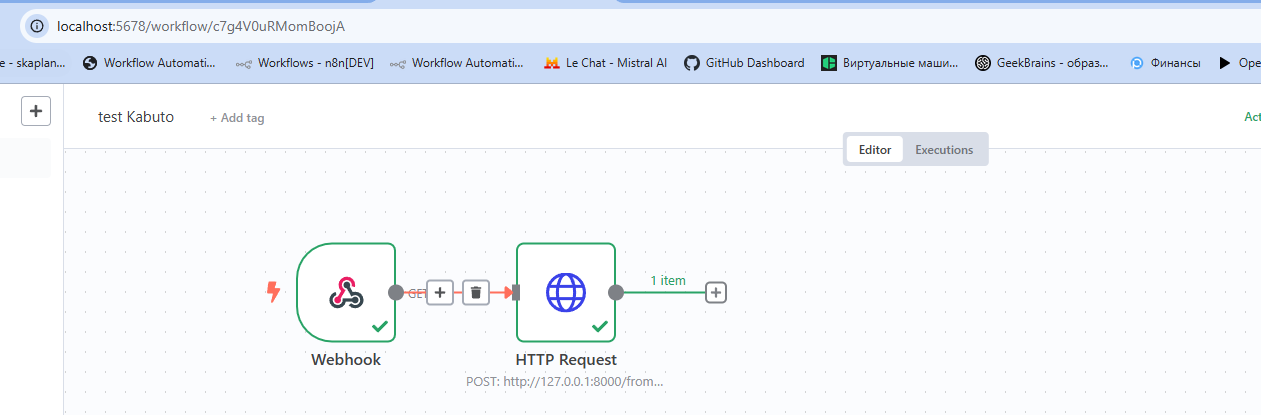
С таким промптом ты бы получил почти в точности тот же бот, который у нас сейчас 👍

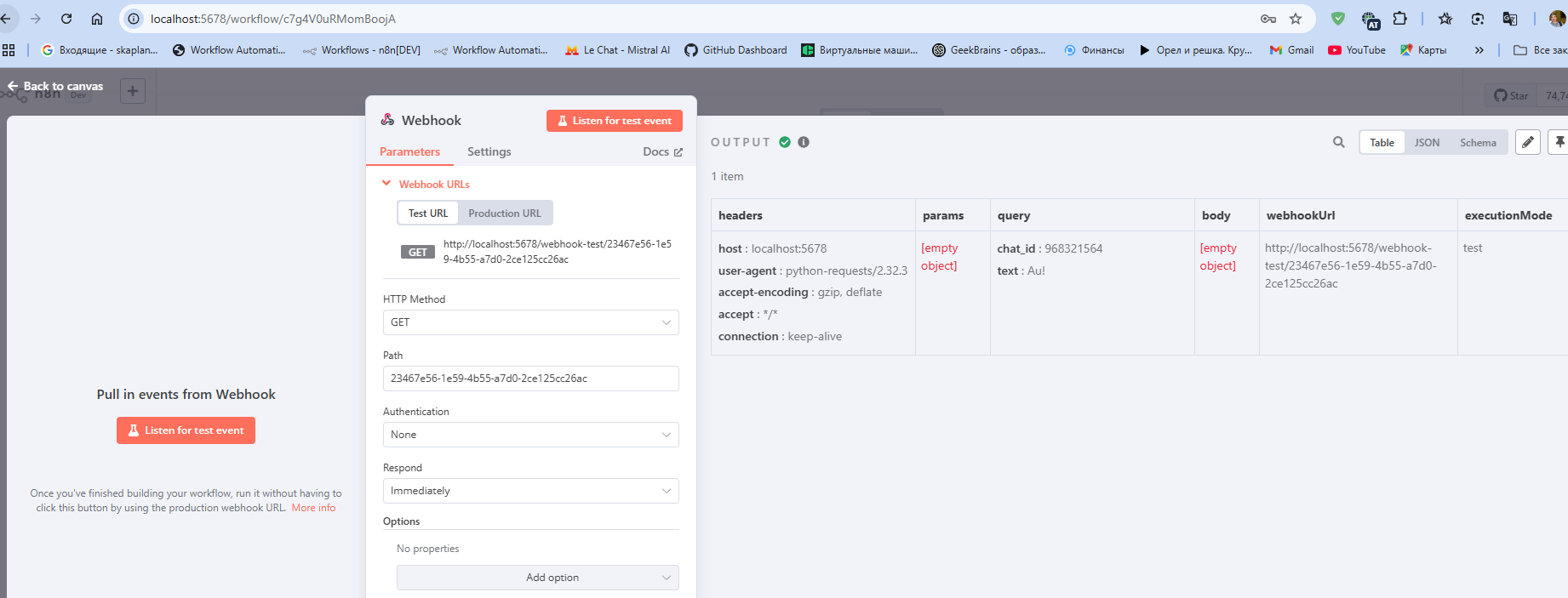
**🔧 Краткая структура того, что ты просил:**

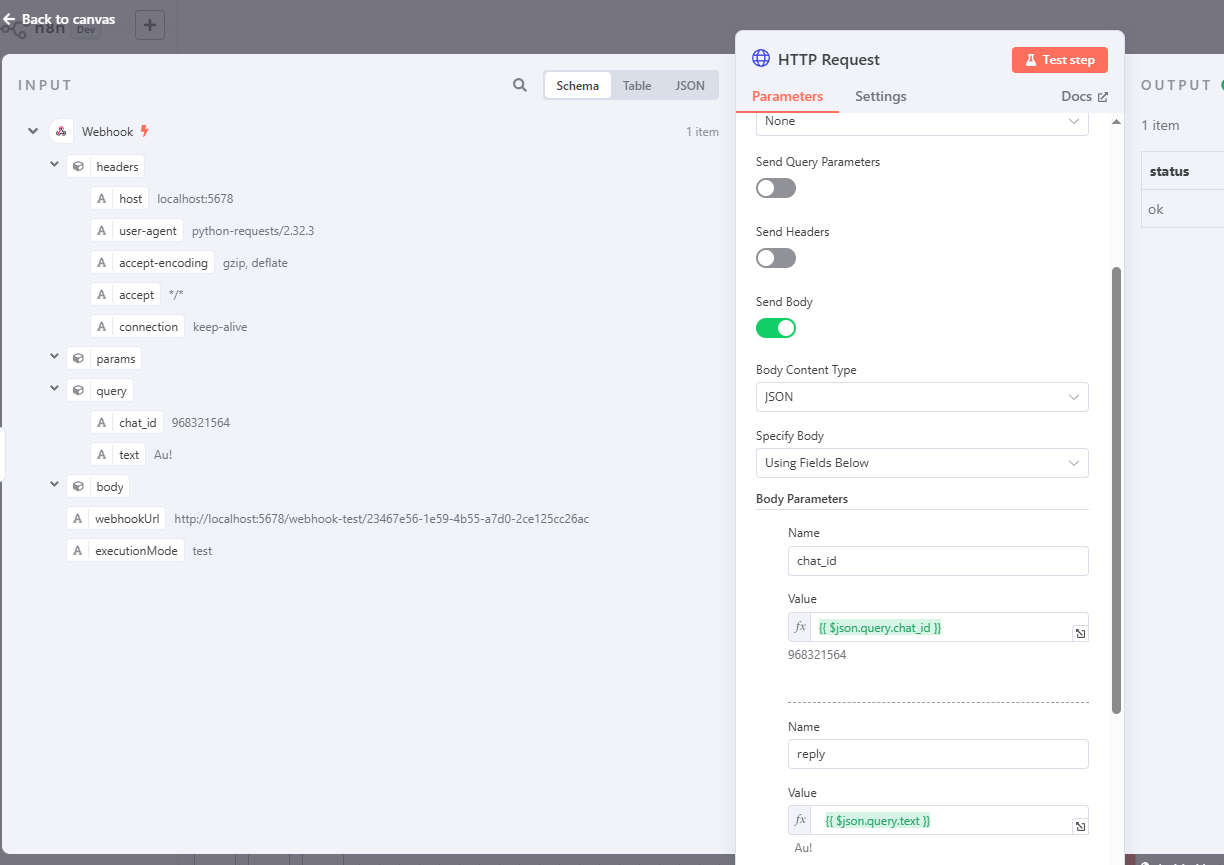
* **aiogram** для приёма сообщений от пользователя.
* **Flask** для получения ответа от n8n.
* **requests.get** для пересылки сообщения из Telegram в n8n.
* **.env** файл для хранения токена безопасно.
* Асинхронная работа Flask и aiogram в одном приложении (через threading.Thread и asyncio loop).

Хочешь — могу сгенерировать по этому промпту свежую версию бота, чтобы ты сравнил?

В n8n создан флоу test Kabuto

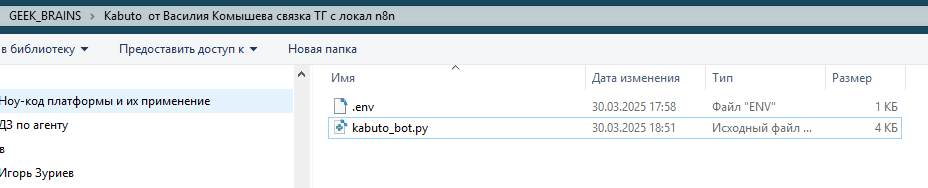






С помощью chatGPT написан бот на пайтон

В VCS запускаем бот –



В Телеграм создан бот

t.me/Ksd\_n8n\_tg\_kabuto\_bot <https://t.me/Ksd_n8n_tg_kabuto_bot>

АПИ бота- 7898022399:AAF6lSDmPOku\_kBQrYdzoj0GM6MxQzKxJRc

Функционал бота – эхо-бот - повторение отправленного сообщения

**Вы сказали:**

Привет! Я запустил бота. Ноды Webhook и HTTP-request в n8n отработали тоже правильно. В результате я получил в Телеграмме эхо того текста, который набрал в боте. Так и было задумано. Благодарю!

**ChatGPT сказал:**

Привет! Отлично, значит, всё работает, как и задумывалось! 🚀🎉

Если нужно будет что-то доработать (например, добавить логику обработки сообщений или изменить структуру флоу), обращайся! 😊

**Вы сказали:**

Представь, что ты был преподаватель, который рассказывает студентам про связку N8n с Телеграм через наш бот, который мы с тобой написали. Сделай такое объяснение наиболее понятным, отрази в нём функционал каждого элемента, последовательность событий.

**ChatGPT сказал:**

**🚀 Как связать n8n с Telegram через промежуточного бота на Python**

👋 Привет, студенты! Сегодня мы разберём, как соединить **n8n** с **Telegram**, используя промежуточного бота, написанного на **Python**.

Мы рассмотрим:  
✅ Зачем нужен бот в этой связке  
✅ Как работают Webhook и HTTP Request в n8n  
✅ Последовательность работы системы

**📌 Зачем нужен Python-бот между n8n и Telegram?**

Telegram не принимает **HTTP (без HTTPS)** вебхуки, а n8n, работая локально, выдаёт именно http://.  
Наш бот выполняет **две основные функции**:  
1️⃣ **Отправляет сообщения из Telegram в n8n**  
2️⃣ **Принимает ответы от n8n и отправляет их обратно пользователю**

Этот бот — связующее звено, которое позволяет n8n и Telegram "разговаривать" друг с другом.

**📌 Как это работает? (Шаги выполнения)**

**1️⃣ Пользователь пишет сообщение боту в Telegram**

🔹 Бот на **aiogram** получает текст от пользователя.  
🔹 Отправляет этот текст в **n8n** через Webhook.

**2️⃣ n8n обрабатывает запрос**

🔹 **Webhook** получает сообщение от бота.  
🔹 **HTTP Request** передаёт его дальше в сценарий n8n.  
🔹 n8n выполняет нужные операции (например, анализ текста, запрос к API и т.д.).  
🔹 n8n отправляет ответ обратно через HTTP-запрос.

**3️⃣ Бот принимает ответ и отправляет его пользователю**

🔹 Бот ждёт запрос на /from-n8n (Flask).  
🔹 Когда n8n отправляет ответ, бот получает chat\_id и reply.  
🔹 Бот отправляет сообщение обратно пользователю в Telegram.

**📌 Компоненты нашей системы**

💡 **Telegram Bot** – получает сообщения от пользователя и отправляет их в n8n.  
💡 **Flask-сервер** – ждёт ответов от n8n и передаёт их в Telegram.  
💡 **n8n Webhook** – принимает входящие сообщения и обрабатывает их.  
💡 **n8n HTTP Request** – передаёт обработанные данные боту.

**📌 Итоговая последовательность работы**

1️⃣ Пользователь пишет в Telegram-бота.  
2️⃣ Бот пересылает текст в n8n через Webhook.  
3️⃣ n8n обрабатывает сообщение (например, добавляет его в CRM).  
4️⃣ n8n отправляет ответ боту через HTTP-запрос.  
5️⃣ Бот принимает ответ и отправляет его пользователю.

**📌 Вывод**

🔥 Теперь у нас есть полноценная интеграция **n8n ↔ Telegram**, работающая через локального Python-бота.  
📌 Вы можете изменять логику n8n, например:

* Отправлять сообщения в канал
* Использовать AI для генерации ответов
* Добавлять обработку команд

👉 Если есть вопросы, задавайте! 🚀

**Вы сказали:**

Очень доходчиво, супер! Но наш флоу на текущий момент очень простой, в результате просто дублируется исходное сообщение пользователя в Телеграмме - так называемы эхо-бот. Если же нам необходимо добавить в n8n функционал для обработки сообщения пользователя и использования AI для генерации ответов, где нужно размещать такую ноду или ноды? После ноды Вебхук но перед имеющейся нодой HTTP-request, отправляющей ответ боту?

**ChatGPT сказал:**

Спасибо! Рад, что объяснение было понятным! 😊

**🔹 Где разместить AI-обработку в n8n?**

Да, если нам нужно добавить AI-обработку перед отправкой ответа в Телеграм, то **новые ноды должны быть между Webhook и HTTP Request**, который отправляет ответ боту.

**🔹 Как будет выглядеть новый флоу?**

1️⃣ **Webhook (Trigger)** – получает текст от пользователя.  
2️⃣ **AI-модель (например, OpenAI / Mistral / Llama / твой ML-сервис)** – обрабатывает сообщение.  
3️⃣ **Формирование ответа (Set / Function / Code / JSON)** – подготавливает текст ответа.  
4️⃣ **HTTP Request (POST на /from-n8n)** – отправляет готовый ответ Python-боту.

**📌 Какие ноды можно использовать для AI-обработки?**

💡 **OpenAI GPT / Mistral / Llama API** – если используешь облачный AI.  
💡 **Code (JavaScript)** – если нужно вызвать свой локальный AI.  
💡 **Function (JavaScript)** – если нужно обработать текст перед отправкой.  
💡 **HTTP Request** – если AI работает на отдельном сервере (локально или в облаке).

**🔹 Итог: Как обновится работа системы?**

1️⃣ Telegram-бот получает текст и передаёт в n8n.  
2️⃣ n8n отправляет его AI-модели, которая генерирует ответ.  
3️⃣ AI-модель возвращает обработанный текст в n8n.  
4️⃣ n8n передаёт ответ Python-боту.  
5️⃣ Python-бот отправляет его пользователю в Телеграм.

📌 Теперь вместо простого **эхо-бота** у нас будет **бот с AI-генерацией ответов!** 🚀

Какой AI хочешь подключить? GPT, Mistral, Ollama? 😎

**Вы сказали:**

Попробую, сообщу тебе, что получится

**ChatGPT сказал:**

Отлично! 🚀 Попробуй, и если что-то пойдёт не так или появятся вопросы — пиши, разберёмся. Жду новости! 😊

Как всё начиналось –

Привет! Я продолжаю попытки связать Телеграм с локальным n8n. Локальный n8n создаёт вебхук с адресом http, который не устраивает Telegram. Мне преподаватель предложил в качестве решения бота на Пайтон, который должен решить проблему. Вот текст этого бота, что скажешь? - pip install aiogram Flask requests nest\_asyncio

import asyncio

import logging

from threading import Thread

from flask import Flask, request

from aiogram import Bot, Dispatcher, F

from aiogram.types import Message

import requests

BOT\_TOKEN = ""

N8N\_WEBHOOK\_URL = "http://localhost:5678/webhook-test/23467e56-1e59-4b55-a7d0-2ce125cc26ac" #берем url из ноды webhook

FLASK\_PORT = 8000

bot = Bot(token=BOT\_TOKEN)

dp = Dispatcher()

app = Flask(\_\_name\_\_)

loop = asyncio.new\_event\_loop()

asyncio.set\_event\_loop(loop)

@dp.message(F.text)

async def handle\_message(message: Message):

params = {

"chat\_id": message.chat.id,

"text": message.text,

}

try:

response = requests.get(N8N\_WEBHOOK\_URL, params=params)

await message.answer(f"Отправлено в n8n. Ответ: {response.text}")

except Exception as e:

await message.answer(f"Ошибка при отправке в n8n: {e}")

@app.route("/from-n8n", methods=["POST"])

def from\_n8n():

data = request.get\_json()

chat\_id = data.get("chat\_id")

reply = data.get("reply")

if not chat\_id or not reply:

return {"error": "chat\_id и reply обязательны"}, 400

loop.call\_soon\_threadsafe(

asyncio.create\_task,

bot.send\_message(chat\_id=chat\_id, text=reply)

)

return {"status": "ok"}

def run\_flask():

app.run(host="0.0.0.0", port=FLASK\_PORT)

async def run\_bot():

logging.basicConfig(level=logging.INFO)

await dp.start\_polling(bot)

if \_\_name\_\_ == "\_\_main\_\_":

Thread(target=run\_flask, daemon=True).start()

loop.run\_until\_complete(run\_bot())

Привет! Код в целом передаёт сообщения между Телеграмом и локальным n8n, но есть несколько проблем:

**Ошибки и недочёты:**

1. **Неправильные отступы в handle\_message**  
   В строке params = { ... } должен быть отступ. Сейчас код выдаст IndentationError.
2. **Некорректный вызов requests.get()**
   * Ты отправляешь GET-запрос с params=params, но n8n webhook обычно принимает POST-запрос с json=data.
   * Если вебхук в n8n настроен только на POST, то GET-запрос просто не будет работать.
3. **Отсутствует обработка ошибок при запуске Flask**
   * Если порт 8000 уже занят, Flask упадёт.
4. **Проблема с loop.call\_soon\_threadsafe()**
   * asyncio.create\_task() нельзя вызывать так в старых версиях Python, лучше использовать loop.create\_task().

**Улучшенный код:**

python

КопироватьРедактировать

import asyncio

import logging

from threading import Thread

from flask import Flask, request

from aiogram import Bot, Dispatcher, F

from aiogram.types import Message

import requests

import nest\_asyncio # Нужно для работы Flask и aiogram в одном потоке

nest\_asyncio.apply()

BOT\_TOKEN = "YOUR\_BOT\_TOKEN"

N8N\_WEBHOOK\_URL = "http://localhost:5678/webhook-test/23467e56-1e59-4b55-a7d0-2ce125cc26ac"

FLASK\_PORT = 8000

bot = Bot(token=BOT\_TOKEN)

dp = Dispatcher()

app = Flask(\_\_name\_\_)

loop = asyncio.new\_event\_loop()

asyncio.set\_event\_loop(loop)

@dp.message(F.text)

async def handle\_message(message: Message):

params = {

"chat\_id": message.chat.id,

"text": message.text,

}

try:

response = requests.post(N8N\_WEBHOOK\_URL, json=params) # POST, а не GET

await message.answer(f"Отправлено в n8n. Ответ: {response.text}")

except Exception as e:

await message.answer(f"Ошибка при отправке в n8n: {e}")

@app.route("/from-n8n", methods=["POST"])

def from\_n8n():

data = request.get\_json()

chat\_id = data.get("chat\_id")

reply = data.get("reply")

if not chat\_id or not reply:

return {"error": "chat\_id и reply обязательны"}, 400

loop.create\_task(bot.send\_message(chat\_id=chat\_id, text=reply))

return {"status": "ok"}

def run\_flask():

try:

app.run(host="0.0.0.0", port=FLASK\_PORT)

except Exception as e:

print(f"Ошибка запуска Flask: {e}")

async def run\_bot():

logging.basicConfig(level=logging.INFO)

await dp.start\_polling(bot)

if \_\_name\_\_ == "\_\_main\_\_":

Thread(target=run\_flask, daemon=True).start()

loop.run\_until\_complete(run\_bot())

### Основные исправления:

✅ **Использую requests.post()**, так как n8n обычно принимает данные через POST  
✅ **Исправлены отступы** в handle\_message  
✅ **Добавил nest\_asyncio.apply()**, чтобы избежать конфликтов между Flask и aiogram  
✅ **Использую loop.create\_task()**, а не call\_soon\_threadsafe()  
✅ **Добавил обработку ошибки при запуске Flask**

Теперь этот бот сможет корректно работать как "мост" между Телеграмом и локальным n8n. 🚀