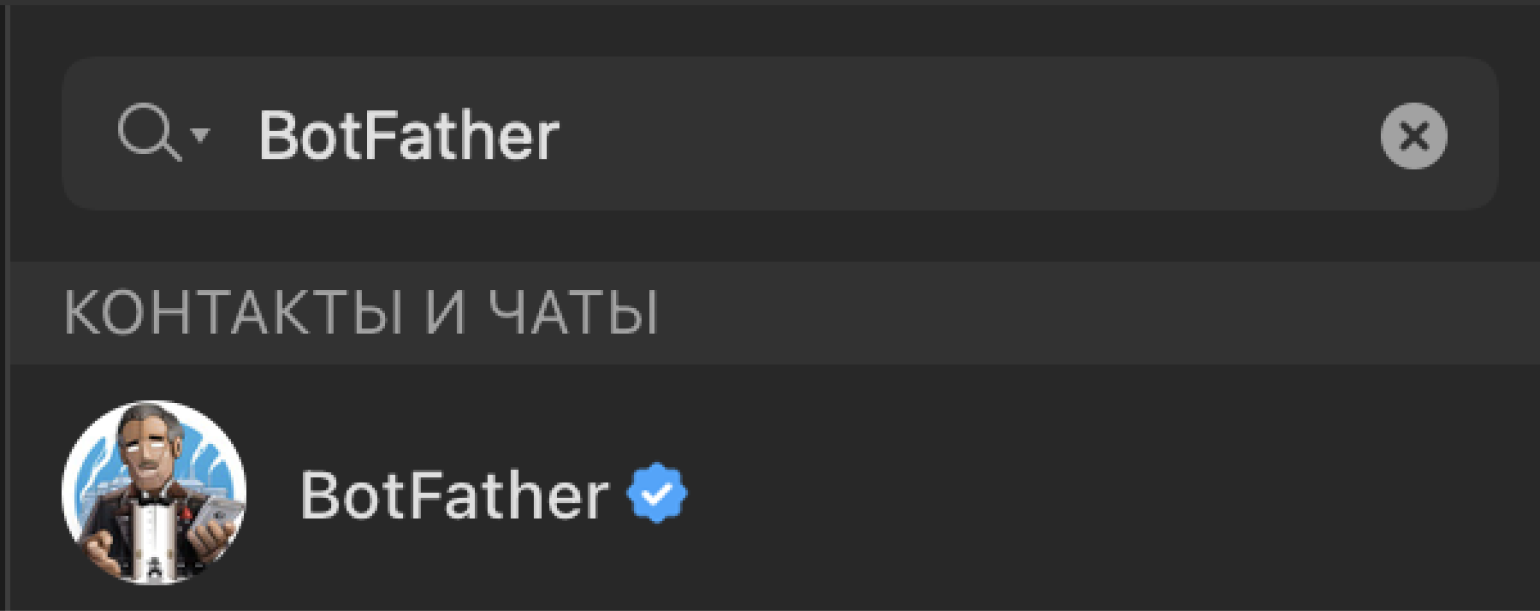
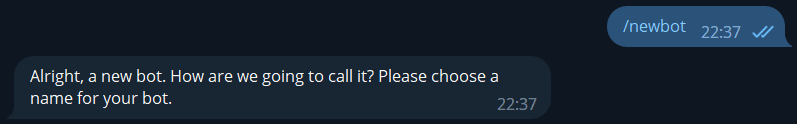
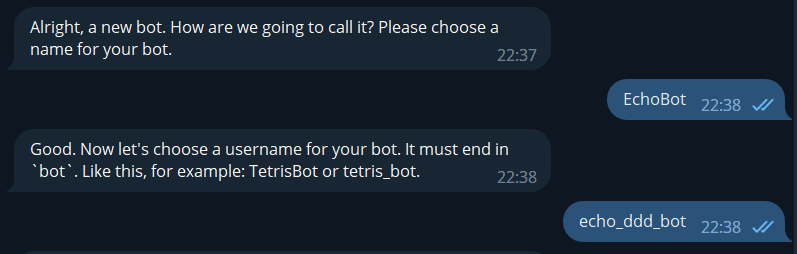
1. Находим в Telegram бота BotFather.



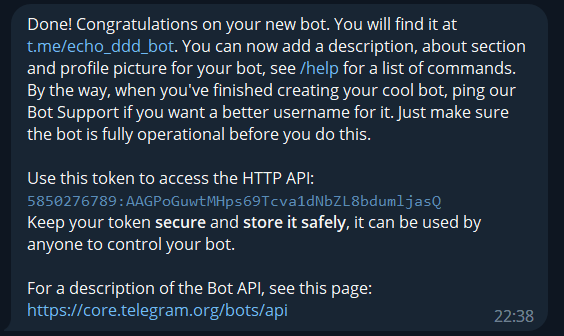
2. Открываем его, жмём кнопку «Запустить» и вводим команду /newbot.



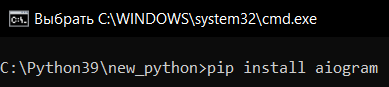
3. Надо придумать название и имя для бота.



4. Сохраняем API токен.



5. При разработке на языке Python если ранее не была установлена библиотека aiogram требуется ее установить через консоль командой pip install aiogram (также должен быть установлен интерпретатор Python).



6. Выбираем удобную среду разработки, создаем файл test.py

7. Подключаем библиотеку и импортируем классы Bot, Dispatcher, executor и модуль types.

Bot определяет, на какие команды от пользователя и каким способом отвечать;

Dispatcher позволяет отслеживать обновления;

Executor запускает бота и выполняет функции, которые следует выполнить.

Модуль types мы будем использовать для команды types.Message, позволяющей работать с текстовыми сообщениями пользователя. Подробно об этом можно [прочесть в документации](https://docs.aiogram.dev/en/latest/telegram/types/index.html).



8. Присваиваем переменной значение токена, полученного ранее.



9. Инициализируем объекты bot и Dispatcher, передав первому наш токен.



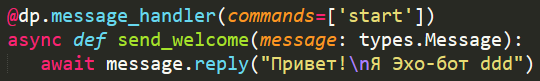
10. Создадим команду для начала работы /start. Создаем message\_handler и прописываем функцию ответа. Также указываем в декораторе, на какую команду реагировать.

message\_handler — это декоратор, который реагирует на входящие сообщения и содержит в себе функцию ответа.

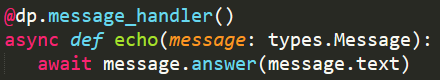
commands=['start'] — это команда, которая связана с декоратором и запускает вложенную в него функцию;

async def send\_welcome — создаёт асинхронную функцию, которая принимает в себя сообщение пользователя message, определяемое через тип Message.

await message.reply — определяет ответ пользователя, используя await из-за асинхронности работы библиотеки.



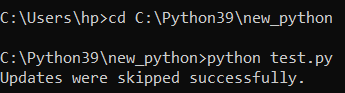
11. Создаем событие, которое будет обрабатывать введённое пользователем сообщение. @dp.message\_handler() создает новое событие, которое запускается в ответ на любой текст, введённый пользователем. Для ответа мы также используем метод message, указывая, что возвращаем исходный текст, принятый в message.



12. Настраиваем получение сообщений от сервера в Telegram. Реализовать получение новых сообщений можно с помощью поллинга. Он работает очень просто — метод start\_polling опрашивает сервер, проверяя на нём обновления. Если они есть, то они приходят в Telegram. Для включения поллинга необходимо добавляем следующие строчки:



13. Сохраняем файл и запускаем код, а затем и бота. Запустить можно через консоль или IDE, если она это позволяет. Если запуск происходит через консоль, то в ней переходим в нужную директорию с файлом и запускаем его при помощи команды python.



14. Проверяем работу бота.



Полный код программы:

from aiogram import Bot, Dispatcher, executor, types

API\_TOKEN = '5850276789:AAGPoGuwtMHps69Tcva1dNbZL8bdumljasQ'

bot = Bot(token=API\_TOKEN)

dp = Dispatcher(bot)

@dp.message\_handler(commands=['start'])

async def send\_welcome(message: types.Message):

await message.reply("Привет!\nЯ Эхо-бот ddd")

@dp.message\_handler()

async def echo(message: types.Message):

await message.answer(message.text)

if \_\_name\_\_ == '\_\_main\_\_':

executor.start\_polling(dp, skip\_updates=True)