**Інформаційні мережі та телекомінікації**

**Лабораторна робота №4.**

**Тема:** Взаємодія з користувачем: inline, deep linking, attachment menu

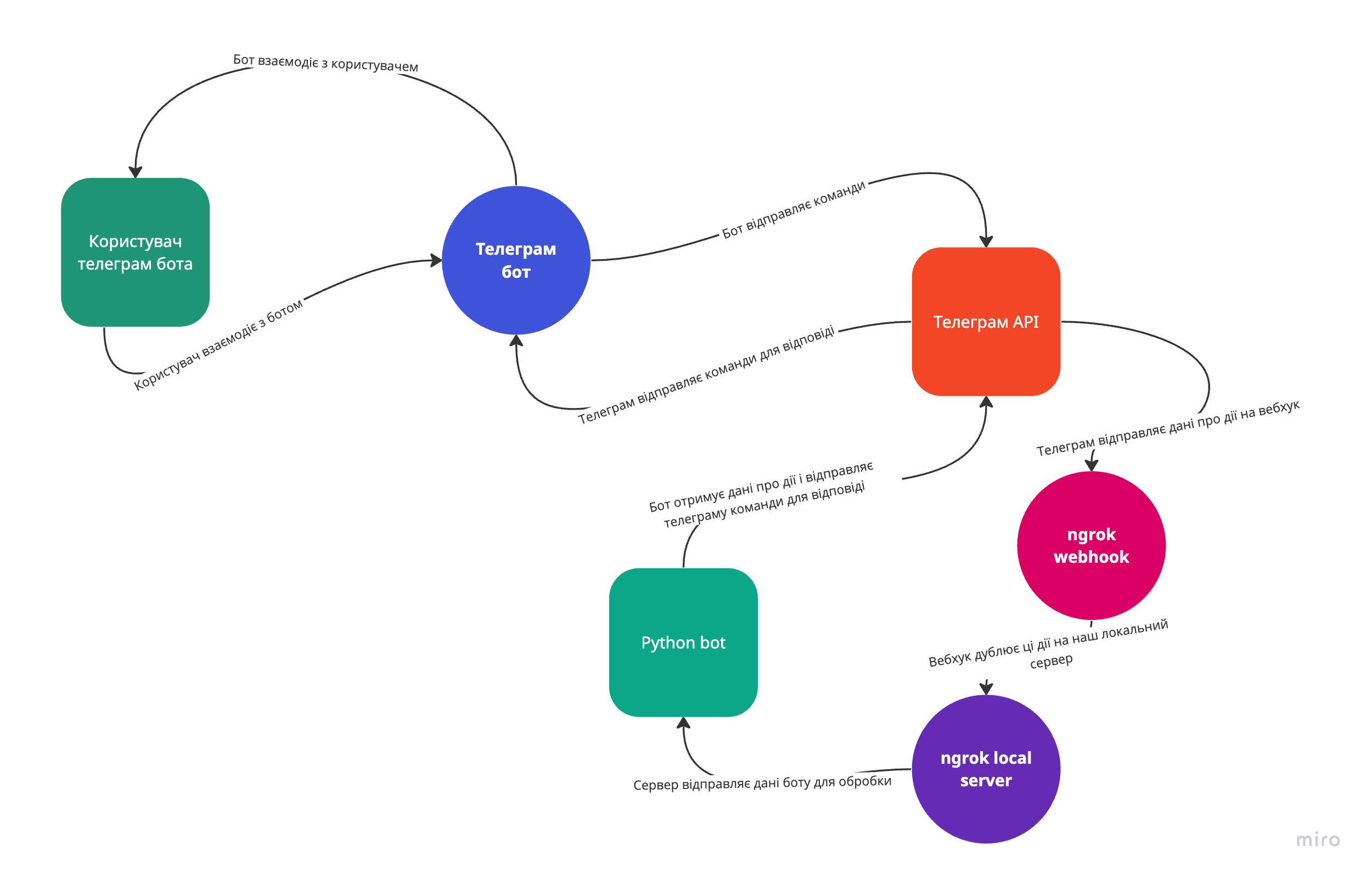
**Мета:** Навчитися налаштовувати і використовувати Telegram Bot через python.

**Хiд роботи**

**І. Виконайте наступні завдання** (у файлі звіту опишіть відповідні процеси):

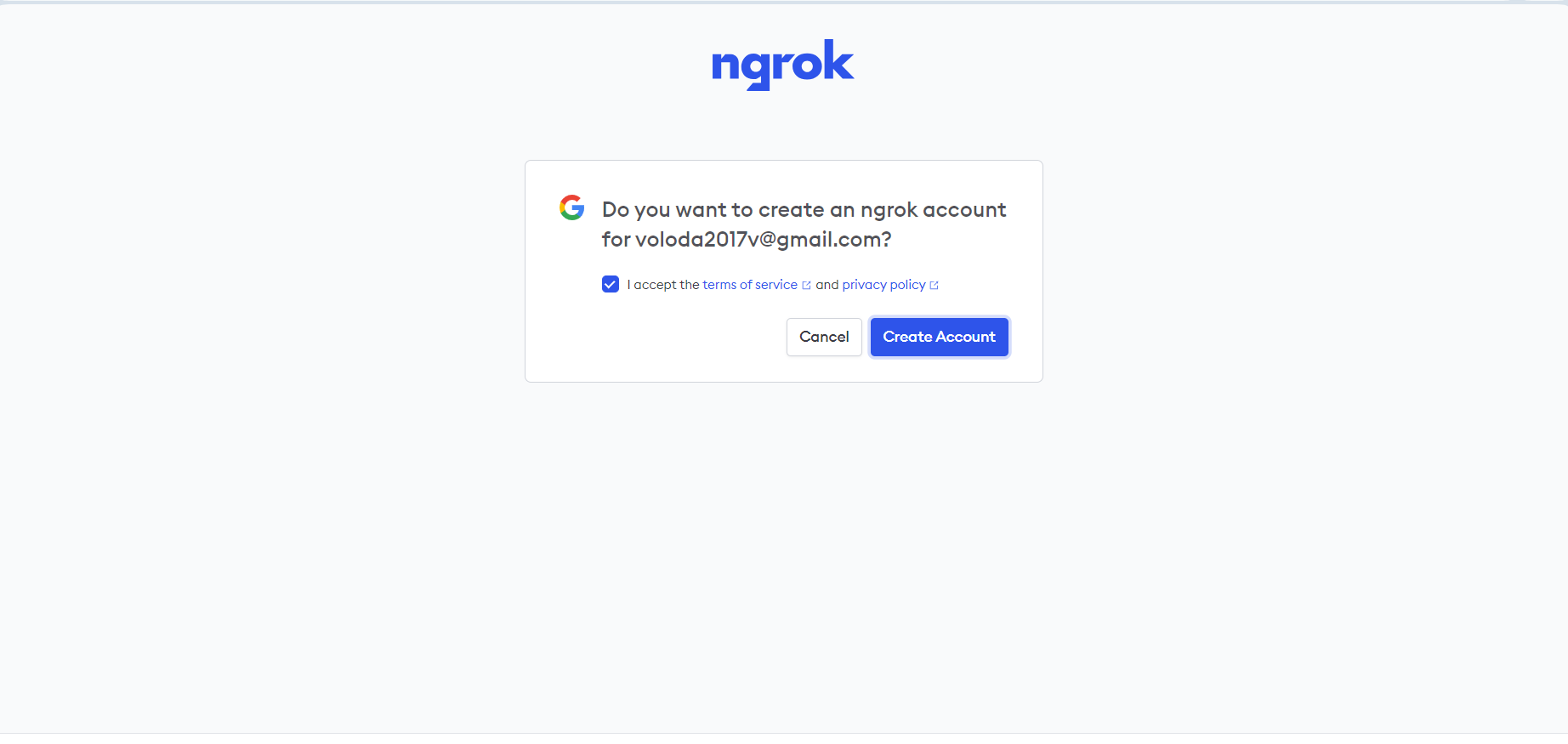
Продовжимо розробку telegram bot на python. В минулій лабораторній ми встановили python та підготували проект по схемі до створення вебхука, для отримання даних з telegram api. В цій лабораторній продовжимо і завершимо цю схему.

Нижче представлена загальна схема побудови бота і взаємодії системи.



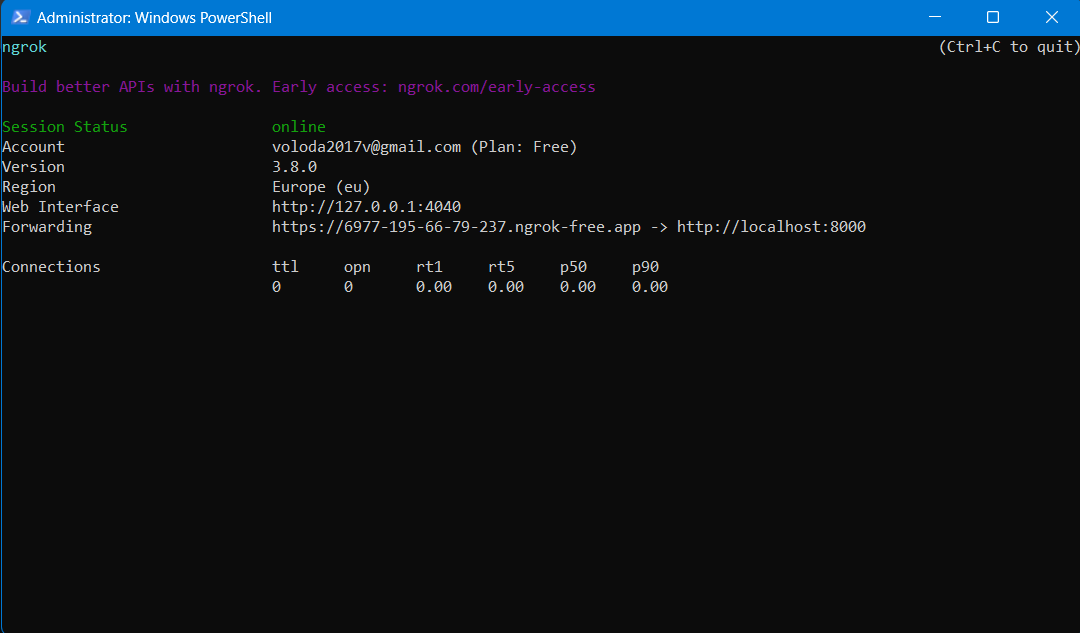
Результат виконання цієї лабораторної закомітьте в гілку lab4 вашого проекту.

**Завдання 1.** Наступний етап: слухати що отримує телеграм від бота і реагувати на зміни. Для цього в телеграм арі є механізм вебхуків(webhooks). Як тільки телеграм отримає якесь повідомлення до бота, він відразу відправить ці дані за посиланням, налаштованим як основний вебхук. А наше завдання: створити в інтернеті таке посилання і слухати, що саме присилає нам телеграм. Для цього ми використаємо **ngrok, який якраз і дозволяє створити посилання в інтернеті і направити його на вашу систему для обробки всіх вхідних даних. Встановіть** <https://ngrok.com/download>, створіть собі аккаунт а збережіть в системі токен командою



ngrok config add-authtoken 2e0icsHMjqvtSnqrqoKqFHF3cBp\_6D5BaAf2coVxh7j2FNkvq

Після цього запустіть ngrok сервер командою

ngrok http 8000 

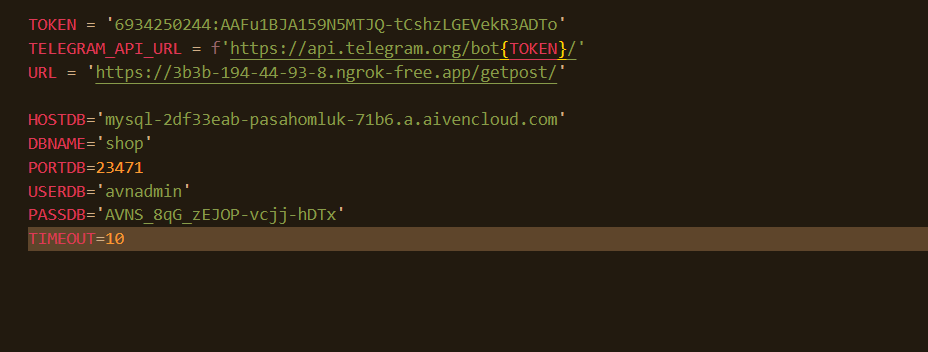
Це дасть нам змогу перенаправляти всі дані, які приходять на сервер ngrok прямо на нашу систему.

**Завдання 2.** Створивши webhook і перенапрямивши його на нашу систему тепер треба налаштувати наш python bot на прослуховування цих даних і генерацію відповідей.

Необхідно зберегти змінну webhook для того, щоб наш бот її міг використовувати: додамо у файл #myapp/credentials.py

Нову змінну URL, яка буде містити складатися з: наш webhook + /getpost/

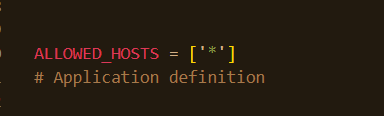
#new, do not forget to change it  
URL = 'https://your\_ngrok\_url.ngrok.io/getpost/'



Далі у файлі #myproject/settings.py

Надамо нашому проекті права отримувати дані з будь-яких джерел

#myproject/settings.py  
  
ALLOWED\_HOSTS = ['\*']



Тепер, коли ми маємо webhook і він перенаправляє дані на нашу систему, нам необхідно вказати для телеграму, щоб він відправляв всі дані на цей вебхук. Зробимо це, використовуючи запит setWebhook

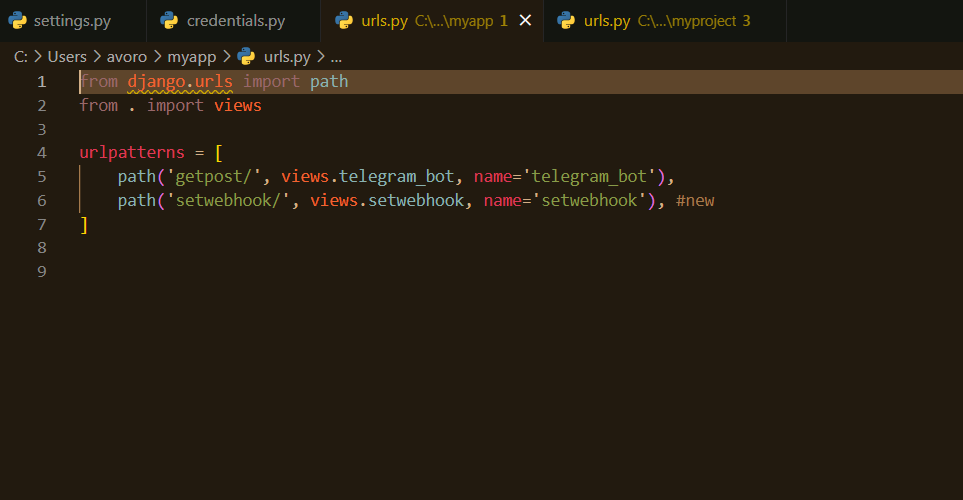
Відкриємо файл **myapp/views.py** та додамо такі налаштування

#myapp/views.py  
  
from myapp.credentials import TELEGRAM\_API\_URL, URL  
  
def setwebhook(request):  
 response = requests.post(TELEGRAM\_API\_URL+ "setWebhook?url=" + URL).json()  
 return HttpResponse(f"{response}")

Після чого створимо новий шлях в налаштуваннях Django

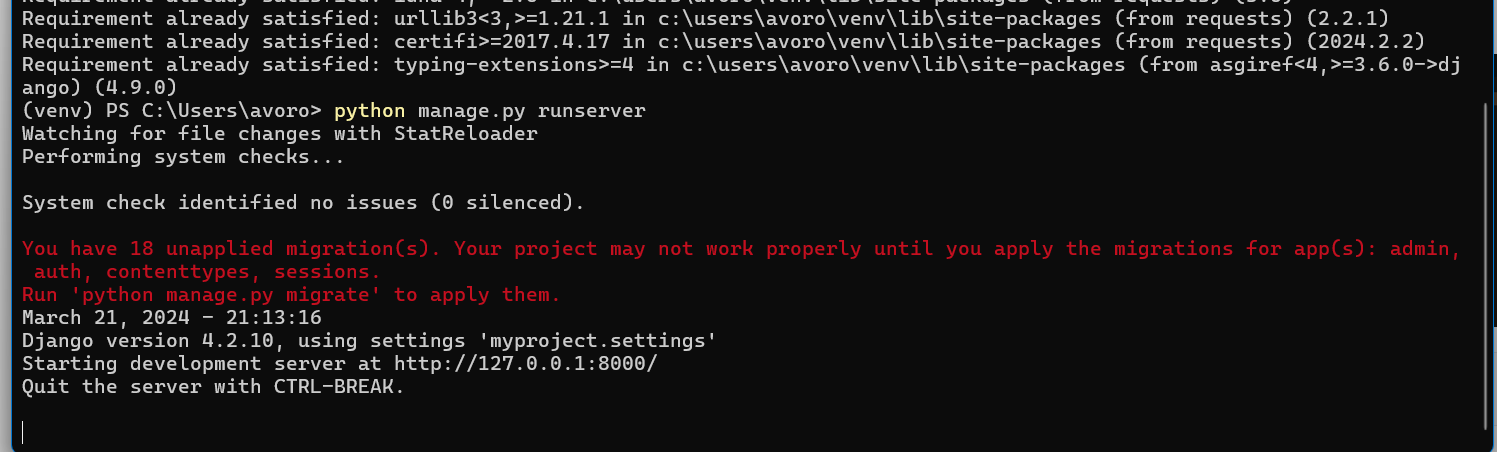
У файлі **myapp/urls.py**

#myapp/urls.py  
  
from django.urls import path  
from . import views  
  
urlpatterns = [  
 path('getpost/', views.telegram\_bot, name='telegram\_bot'),  
 path('setwebhook/', views.setwebhook, name='setwebhook'), #new  
]



Готово. Створивши цю логіку ми можемо запустити наш скрипт

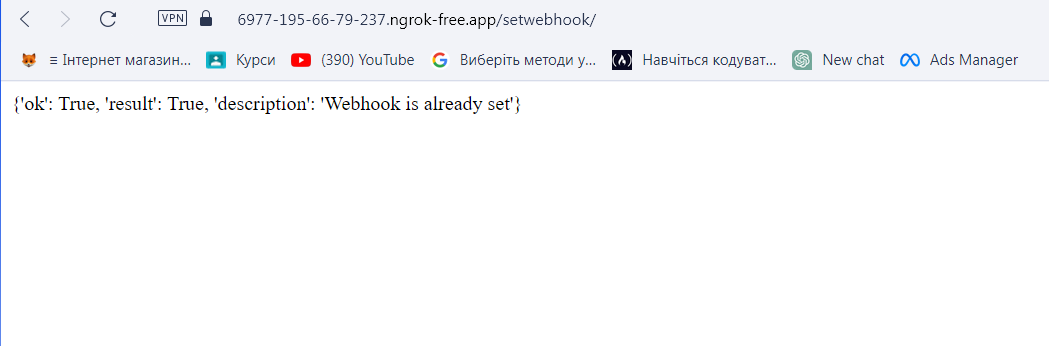
python manage.py runserver

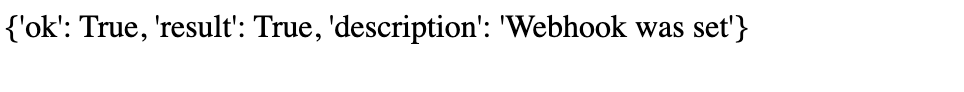


Тепер, для того щоб виконати запит setWebhook нам необхідно просто відкрити в браузері це посилання <https://your_ngrok_url.ngrok.io/'setwebhook/>

(ЗМІНИТИ ЧАСТИНУ ПОСИЛАННЯ НА ВАШ ВЕБХУК) і ми відправимо в телеграм апі запит на збереження інформації про вебхук. Цей самий крок можна було виконати через postman, відправивши аналогічний запит https://core.telegram.org/bots/api#setwebhook

Результатом правильного виконання повинна бути відповідь у браузері:





Завдяки цього кроку ми зробили автоматичну відправку з telegram всіх даних, які буде отримувати наш сервер на webhook ngrok, який в свою чергу перенаправляє всі ці дані на нашу систему. Тепер ми можемо реагувати на ці дані і відповідати зі сторони бота.

**Завдання 3.** Маючи всю необхідну інфраструктуру тепер ми можемо відповідати з сторони бота на повідомлення, які відправляють йому користувачі. Давайте напишемо відповідь на повідомлення, повертаючи той самий текст, який користувач пише боту.

Для цього нам необхідно відредагувати файл #myapp/views.py

Додати логіку відправи повідомлення через запит sendMessage

import json  
import requests  
from myapp.credentials import TELEGRAM\_API\_URL, URL  
from django.http import HttpResponse, HttpResponseBadRequest  
from django.views.decorators.csrf import csrf\_exempt  
  
def setwebhook(request):  
 response = requests.post(TELEGRAM\_API\_URL+ "setWebhook?url=" + URL).json()  
 return HttpResponse(f"{response}")  
  
@csrf\_exempt  
def telegram\_bot(request):  
 if request.method == 'POST':  
 update = json.loads(request.body.decode('utf-8'))  
 handle\_update(update)  
 return HttpResponse('ok')  
 else:  
 return HttpResponseBadRequest('Bad Request')  
  
def handle\_update(update):  
 chat\_id = update['message']['chat']['id']  
 text = update['message']['text']  
 send\_message("sendMessage", {  
 'chat\_id': chat\_id,  
 'text': f'you said {text}'  
 })  
  
def send\_message(method, data):  
 return requests.post(TELEGRAM\_API\_URL + method, data)

Ми підписалися на всі дані, які отримує бот, і якщо це POST запит, відправляємо повідомлення у відповідь з даними, які відправив користувач, використовуючи запит sendMessage

Перевірте, чи працює наш бот і чи повертає телеграм бот у відповідь ваше повідомлення.

Реалізувати відправку повідомлення також можна було б через postman і запит <https://core.telegram.org/bots/api#sendmessage> Для цього необхідно знати chat\_id користувача, якому необхідно відправити повідомлення. Ви його дізнавалися в минулій лабораторній, коли писали повідомлення боту і робити запит на getUpdates. 