## **Державний університет інформаційно-комунікаційних технологій**

**Практична робота №4**

**з дисципліни: Технології штучного інтелекту**

**Тема: Розробка чат-ботів на Руоп з використанням бібліотеки Aiogram 3.0**

Виконав:  
студент ДУІКТ  
Тертишний В.Ю.

група: ШІДМ-51

м.Київ

**Мета:** Набуття теоретичних знань та практичних навичок в розробці чат-ботів для платформи Теісртат з використанням бібліотеки Aiogram 3.0 на Python; проектування ефективних і функціональних чат-ботів, з використанням принципів асинхронної обробки запитів, інтеграцією чат-ботів з базами даних та платіжними системами.

**Код:**

import asyncio

import logging

import sqlite3

import requests

from aiogram import Bot, Dispatcher, F

from aiogram.filters import Command

from aiogram.types import (

Message,

CallbackQuery,

ContentType,

InlineKeyboardButton,

InlineKeyboardMarkup,

ReplyKeyboardMarkup,

KeyboardButton,

)

from aiogram.utils.keyboard import InlineKeyboardBuilder, ReplyKeyboardBuilder

from dotenv import load\_dotenv

import os

load\_dotenv()

# ------------------ Налаштування логування ------------------ #

logging.basicConfig(

level=logging.INFO,

format="%(asctime)s [%(levelname)s] %(name)s: %(message)s",

)

# ------------------ Ваш токен бота ------------------ #

BOT\_TOKEN = os.getenv('TOKEN')

# ------------------ Приклад API-ключа для OpenWeather (необов'язково) ------------------ #

WEATHER\_API\_KEY = os.getenv('WEATHER\_API\_KEY')

# ------------------ Налаштування / створення SQLite БД ------------------ #

conn = sqlite3.connect("users.db")

cursor = conn.cursor()

cursor.execute("""

CREATE TABLE IF NOT EXISTS users (

id INTEGER PRIMARY KEY AUTOINCREMENT,

user\_id INTEGER NOT NULL,

username TEXT,

full\_name TEXT,

favorite\_brand TEXT,

favorite\_color TEXT

);

""")

conn.commit()

async def cmd\_start(message: Message) -> None:

"""

Обробник команди /start

"""

user\_id = message.from\_user.id

username = message.from\_user.username

full\_name = message.from\_user.full\_name

# Перевіримо, чи існує користувач у БД

cursor.execute("SELECT \* FROM users WHERE user\_id = ?", (user\_id,))

user = cursor.fetchone()

# Якщо немає – створимо запис

if not user:

cursor.execute(

"INSERT INTO users (user\_id, username, full\_name) VALUES (?, ?, ?)",

(user\_id, username, full\_name),

)

conn.commit()

await message.answer(

"Привіт! Я бот для підбору квадроцикла.\n"

"Надішліть /help, щоб дізнатися, що я вмію."

)

async def cmd\_help(message: Message) -> None:

"""

Обробник команди /help

"""

help\_text = (

"Список команд:\n"

"/start - Почати роботу\n"

"/help - Допомога\n"

"/weather - Перевірити погоду\n"

"/select\_brand - Обрати бренд квадроцикла\n"

"/select\_color - Обрати колір квадроцикла\n"

)

await message.answer(help\_text)

async def cmd\_weather(message: Message) -> None:

"""

Приклад отримання погоди через OpenWeatherMap

"""

city = "Kyiv"

url = (

f"https://api.openweathermap.org/data/2.5/weather?"

f"q={city}&appid={WEATHER\_API\_KEY}&units=metric&lang=ua"

)

try:

response = requests.get(url)

data = response.json()

# Перевіримо код відповіді

if data.get("cod") != 200:

await message.answer("Не вдалося отримати інформацію про погоду.")

return

temp = data["main"]["temp"]

desc = data["weather"][0]["description"]

await message.answer(

f"Погода в {city} зараз:\n"

f"Температура: {temp}°C\n"

f"Опис: {desc}"

)

except Exception as e:

logging.exception("Помилка при отриманні погоди:")

await message.answer("Сталася помилка при отриманні погоди.")

async def cmd\_select\_brand(message: Message) -> None:

"""

Приклад використання Reply-клавіатури для вибору бренда

"""

kb\_builder = ReplyKeyboardBuilder()

brands = ["Polaris", "Can-Am", "Yamaha", "Honda"]

for brand in brands:

kb\_builder.button(text=brand)

kb\_builder.adjust(2) # 2 кнопки в один ряд

keyboard = kb\_builder.as\_markup(resize\_keyboard=True, one\_time\_keyboard=True)

await message.answer("Оберіть бренд квадроцикла:", reply\_markup=keyboard)

async def cmd\_select\_color(message: Message) -> None:

"""

Приклад використання Inline-кнопок для вибору кольору

"""

kb\_builder = InlineKeyboardBuilder()

colors = ["Червоний", "Синій", "Зелений", "Чорний"]

for color in colors:

cb\_data = f"color\_{color.lower()}"

kb\_builder.button(

text=color,

callback\_data=cb\_data

)

kb\_builder.adjust(2)

await message.answer(

"Оберіть колір квадроцикла:",

reply\_markup=kb\_builder.as\_markup()

)

async def callback\_color(call: CallbackQuery) -> None:

"""

Обробка натискання інлайн-кнопок кольору

"""

# Витягнемо колір

raw\_color = call.data.replace("color\_", "")

color = raw\_color.capitalize()

# Збережемо у БД

user\_id = call.from\_user.id

cursor.execute(

"UPDATE users SET favorite\_color = ? WHERE user\_id = ?",

(color, user\_id)

)

conn.commit()

# Відповідь користувачу

await call.message.answer(f"Ви обрали колір: {color}")

await call.answer() # Закрити "годинник" на кнопці

async def text\_handler(message: Message) -> None:

"""

Загальна обробка будь-яких інших текстових повідомлень

"""

user\_text = message.text.strip()

# Перевіримо, чи це один із брендів

possible\_brands = ["Polaris", "Can-Am", "Yamaha", "Honda"]

if user\_text in possible\_brands:

user\_id = message.from\_user.id

cursor.execute(

"UPDATE users SET favorite\_brand = ? WHERE user\_id = ?",

(user\_text, user\_id)

)

conn.commit()

await message.answer(f"Ви обрали бренд: {user\_text}")

else:

# Якщо це не бренд, то просто відповідаємо загальним текстом

await message.answer(

"Я отримав ваше повідомлення, але не зовсім розумію.\n"

"Спробуйте обрати команду чи скористатися меню."

)

async def photo\_handler(message: Message) -> None:

"""

Обробка отриманого фото

"""

# Можна зберегти його file\_id, скачати чи виконати інші дії.

await message.answer("Дякую за фото! Я можу зберегти його для подальшої обробки.")

async def main():

# Створюємо бот і диспетчер

bot = Bot(token=BOT\_TOKEN)

dp = Dispatcher()

# Реєструємо хендлери для команд

dp.message.register(cmd\_start, Command(commands=["start"]))

dp.message.register(cmd\_help, Command(commands=["help"]))

dp.message.register(cmd\_weather, Command(commands=["weather"]))

dp.message.register(cmd\_select\_brand, Command(commands=["select\_brand"]))

dp.message.register(cmd\_select\_color, Command(commands=["select\_color"]))

# Обробка inline callback

dp.callback\_query.register(callback\_color, F.data.startswith("color\_"))

# Обробка фото

dp.message.register(photo\_handler, F.content\_type == ContentType.PHOTO)

# Обробка будь-якого іншого тексту (зверніть увагу, що має бути вкінці)

dp.message.register(text\_handler, F.text)

try:

logging.info("Бот запущено. Очікуємо повідомлень...")

await dp.start\_polling(bot)

except Exception as err:

logging.exception("Помилка при запуску бота:")

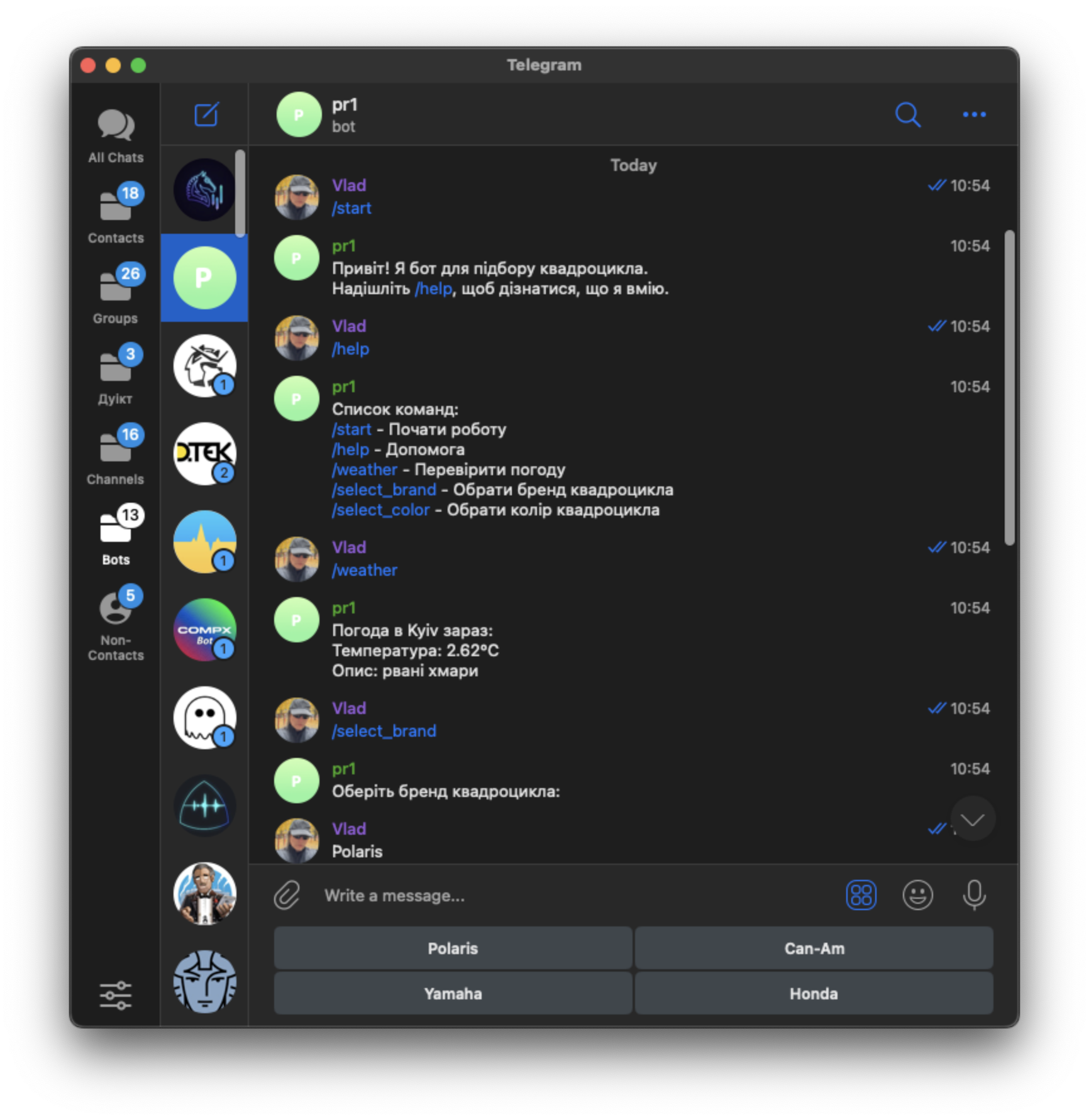
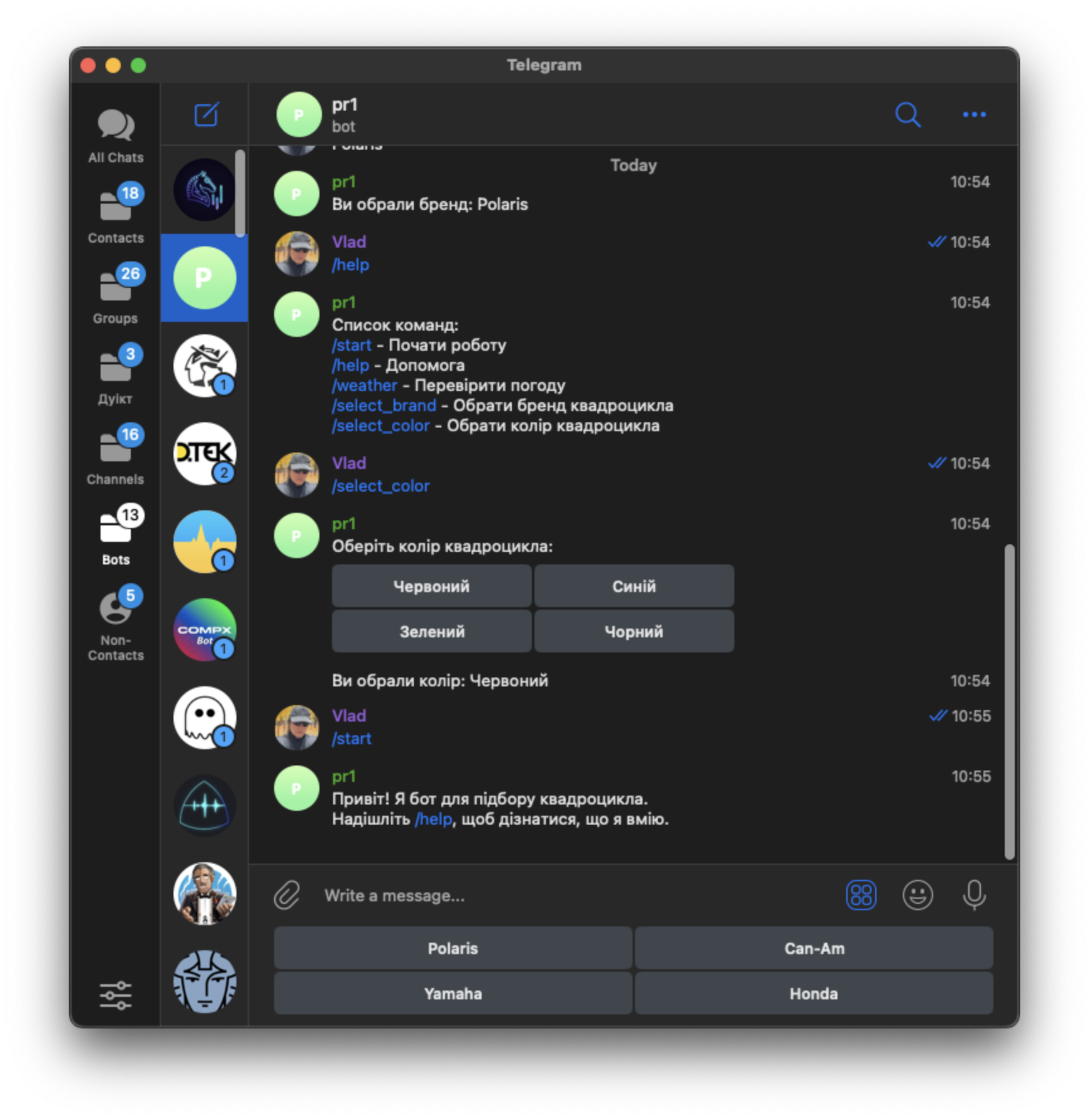
finally:

# Закриваємо підключення до БД при зупинці

conn.close()

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

asyncio.run(main())

**Робота коду:**  
 ****

**Висновки**

У процесі досягнення поставленої мети було здобуто теоретичні знання та практичні навички розробки чат-ботів для платформи Telegram із використанням бібліотеки Aiogram 3.0 на мові програмування Python. Детально опрацьовано принципи асинхронної обробки запитів, що дозволило забезпечити ефективну взаємодію користувачів із ботом без затримок і перевантажень. Особливу увагу приділено проектуванню функціональних чат-ботів із чіткою структурою, адаптованих до виконання завдань різної складності. Розроблено та протестовано механізми інтеграції з базами даних, що забезпечило можливість зберігання і обробки інформації, а також налаштовано інтеграцію з платіжними системами, яка дозволяє реалізувати функції онлайн-оплат. У результаті отримано практичний досвід, який стане основою для подальшого вдосконалення навичок у сфері розробки сучасних чат-ботів.