

## **Практическая работа №2**

### **Основы работы с технологиями контейнеризации и ботами Telegram**

**Цель работы:** создать сервер с постоянно работающим Telegram ботом.

#### **Ход работы**

Наша задача состоит в том, чтобы разобраться, во-первых, с технологией VPN, подключившись к студенческому серверу, во-вторых, создать Telegram бота и собирать docker образ с именем – номером зачетки студента

#### **1. Подключение к серверам**

1.1. Используя Windows PowerShell, устанавливается SSH-соединение с сервером-шлюзом.

1.2. С сервера-шлюза выполняется подключение к основному рабочему серверу.

#### **2. Подготовка рабочего окружения**

2.1. После создается рабочая директория с номером зачетной книжки студента, и выполняется переход в нее.

#### **3. Установка необходимых пакетов**

3.1. Устанавливается библиотека telepot, необходимая для работы с Telegram API.

#### **4. Создание Telegram-бота**

4.1. В Telegram находим и запускаем «**@BotFather**», после создаем нового бота и получаем уникальный токен доступа (рис. 1)

4.2. В каталоге проекта создается Python-скрипт bot.py с кодом бота, после сохраняем и запускаем бота (рис. 1.1):

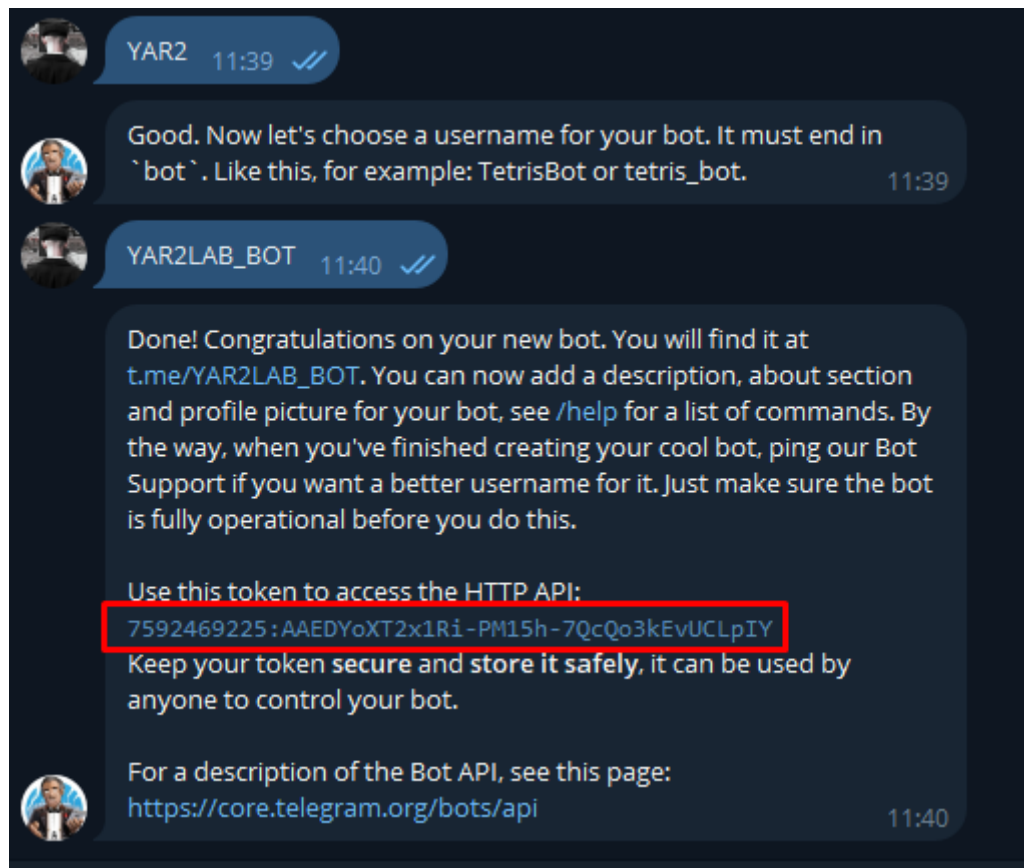


Рисунок 1 – уникальный токен

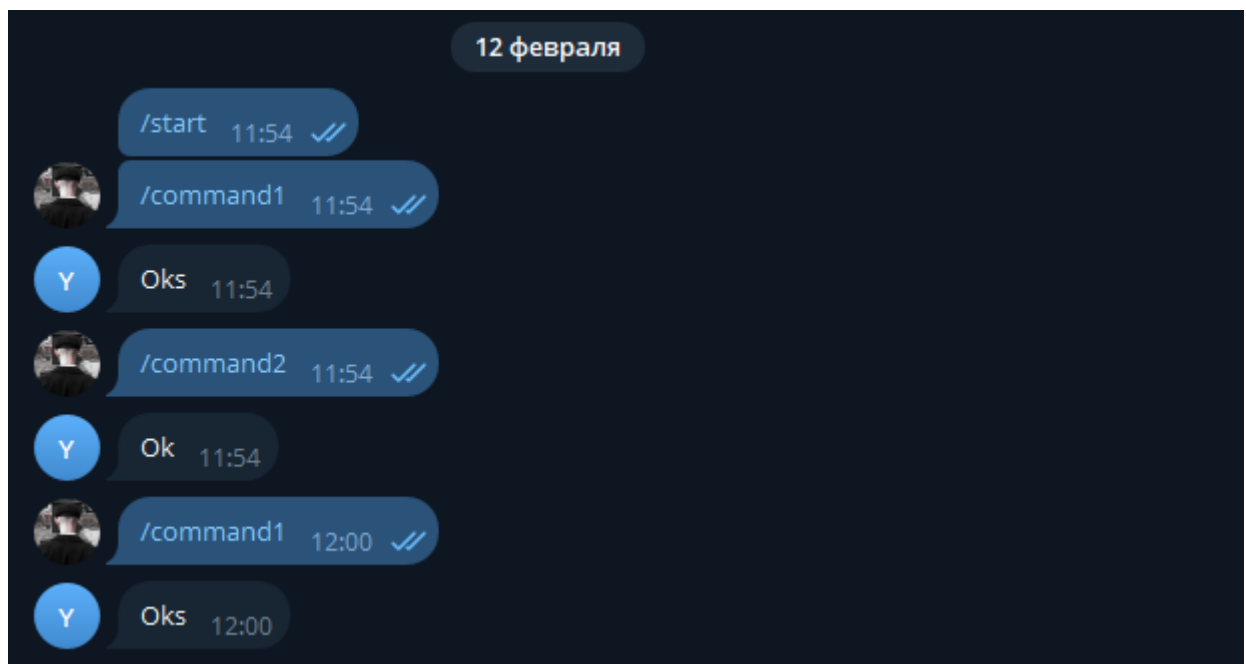


Рисунок 1.1 – работающий бот

Теперь приступим к контейнеризации Python-программы в Docker

## 5. Создание файла requirements.txt

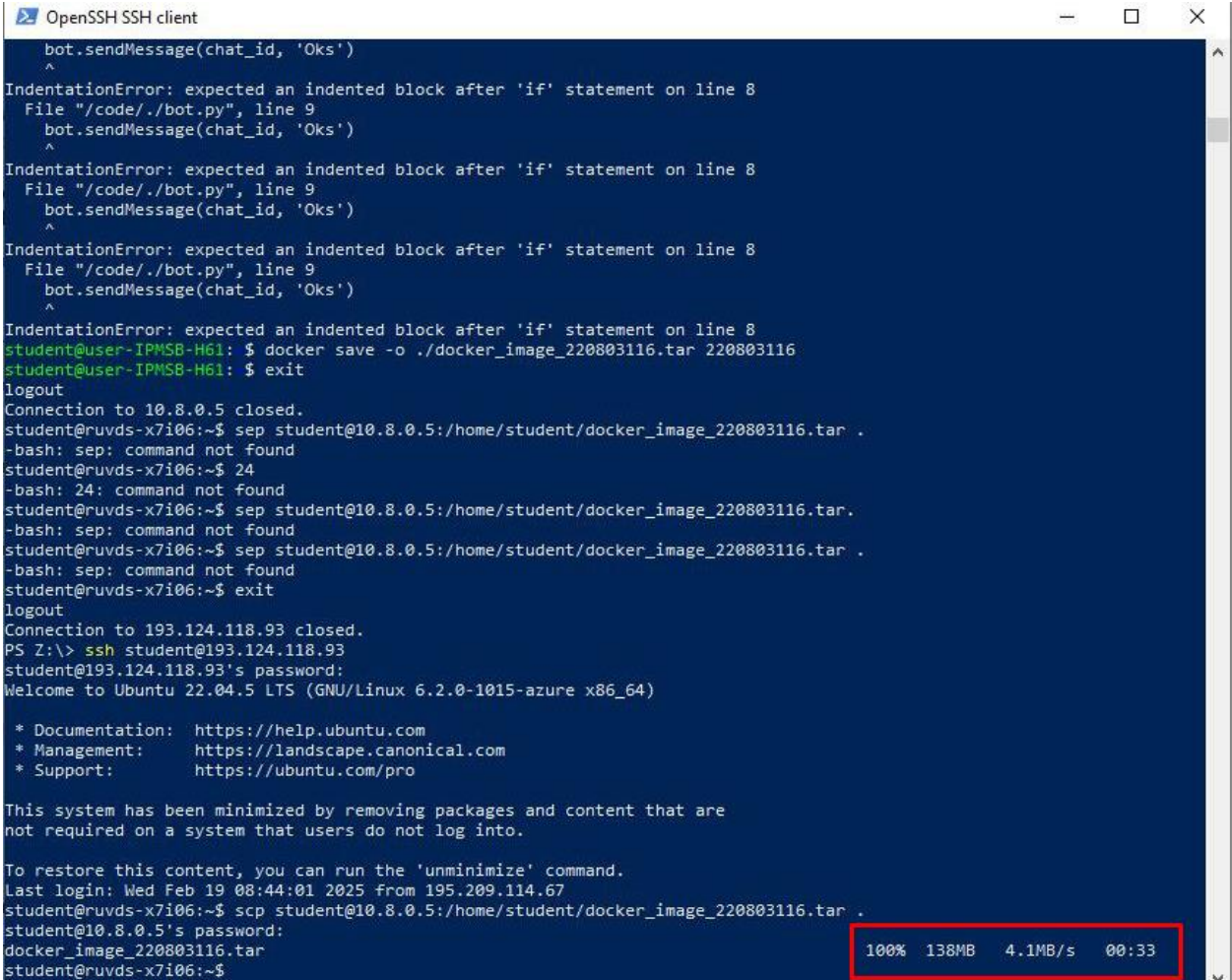
5.1 Этот файл содержит список зависимостей Python, необходимых для работы бота.

5.2 Добавляем telepot, который является библиотекой Python, предназначенная для взаимодействия с Telegram Bot API. После создаем Dockerfile и добавляем код — **Dockerfile**, используемый для создания Docker-образа с нашим Telegram-ботом.

## 6. Сборка Docker-образа

6.1 Собираем образ с номером нашей зачетки, запускаем контейнер с автозапуском, просматриваем списки контейнеров и ищем наш по номеру зачетки.

6.2 После сохраняем Docker-образ на компьютер (рис. 1.2):



```
bot.sendMessage(chat_id, 'Oks')
^
IndentationError: expected an indented block after 'if' statement on line 8
File "/code/./bot.py", line 9
  bot.sendMessage(chat_id, 'Oks')
  ^
IndentationError: expected an indented block after 'if' statement on line 8
File "/code/./bot.py", line 9
  bot.sendMessage(chat_id, 'Oks')
  ^
IndentationError: expected an indented block after 'if' statement on line 8
File "/code/./bot.py", line 9
  bot.sendMessage(chat_id, 'Oks')
  ^
student@user-IPMSB-H61: $ docker save -o ./docker_image_220803116.tar 220803116
student@user-IPMSB-H61: $ exit
logout
Connection to 10.8.0.5 closed.
student@ruvds-x7i06:~$ sep student@10.8.0.5:/home/student/docker_image_220803116.tar .
-bash: sep: command not found
student@ruvds-x7i06:~$ 24
-bash: 24: command not found
student@ruvds-x7i06:~$ sep student@10.8.0.5:/home/student/docker_image_220803116.tar .
-bash: sep: command not found
student@ruvds-x7i06:~$ sep student@10.8.0.5:/home/student/docker_image_220803116.tar .
-bash: sep: command not found
student@ruvds-x7i06:~$ exit
logout
Connection to 193.124.118.93 closed.
PS Z:\> ssh student@193.124.118.93
student@193.124.118.93's password:
Welcome to Ubuntu 22.04.5 LTS (GNU/Linux 6.2.0-1015-azure x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.
Last login: Wed Feb 19 08:44:01 2025 from 195.209.114.67
student@ruvds-x7i06:~$ scp student@10.8.0.5:/home/student/docker_image_220803116.tar .
student@10.8.0.5's password:
docker_image_220803116.tar
student@ruvds-x7i06:~$
```

100% 138MB 4.1MB/s 00:33

Рисунок 1.2 – сохраненный Docker-образ

**Вывод:** в результате практической работы мы научились работать с Telegram ботом, изучили технологию VPN путем подключения к серверу-шлюзу по ssh. А также поработали с Docker файлами.

