

## Отчет по практической работе №2

### Основы работы с технологиями контейнеризации и ботами Telegram

**Цель работы:** Создание сервера с постоянно работающим Telegram ботом.

**Ход работы:**

Для начала необходимо подключиться к серверу шлюзу, который позволит попасть на основной сервер.

```
democran@democran-Modern-15-B5M:~$ ssh student@193.124.118.93
The authenticity of host '193.124.118.93 (193.124.118.93)' can't be established.
ED25519 key fingerprint is SHA256:9Yff0RJ043svji7MPRcbpG4Jp3k2f7tVmDEavUEQwYQ.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '193.124.118.93' (ED25519) to the list of known hosts
.
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: Mon Feb 17 15:38:16 2025 from 194.85.210.24
student@ruvds-x7i06:~$
```

Рисунок 2.1 - Подключение к серверу-шлюзу.

```
student@ruvds-x7i06:~$ ssh student@10.8.0.5
student@10.8.0.5's password:
Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.15.0-131-generic x86_64)

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

Expanded Security Maintenance for Applications is not enabled.

1 update can be applied immediately.
1 of these updates is a standard security update.
To see these additional updates run: apt list --upgradable

12 additional security updates can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm

New release '22.04.5 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Your Hardware Enablement Stack (HWE) is supported until April 2025.
Last login: Mon Feb 17 18:38:43 2025 from 10.8.0.1
student@user-IPMSB-H61:~$
```

Рисунок 2.2 - Подключение к основному серверу.

Следуя методическим указаниям, начинаем работу с файловой системой. Создаем папку с номером зачетной книжки и переходим в папку. После этого проверим работу python на сервере.

```
student@user-IPMSB-H61:~$ mkdir 220803146 && cd 220803146
student@user-IPMSB-H61:~/220803146$ python3.10
Python 3.10.16 (main, Dec 4 2024, 08:53:37) [GCC 9.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> exit()
student@user-IPMSB-H61:~/220803146$
```

Рисунок 2.3 - Создание каталога и проверка работоспособности python.

Для того чтобы не мешать другим пользователям, и не нарушать структуру проекта, создадим окружение. Сразу же активируем его.

```
student@user-IPMSB-H61:~/220803146$ python3.10 -m venv env
student@user-IPMSB-H61:~/220803146$ source env/bin/activate
(env) student@user-IPMSB-H61:~/220803146$
```

Рисунок 2.4 - Создание окружения проекта.

Для работы нашего бота, необходима библиотека telepot версии 12.7. В нашем случае установка определенной версии не будет мешать другим пользователям, с учетом вышеперечисленного пункта.

```
(env) student@user-IPMSB-H61:~/220803146$ pip install telepot==12.7
Collecting telepot==12.7
  Using cached telepot-12.7.tar.gz (73 kB)
  Preparing metadata (setup.py) ... done
Collecting urllib3>=1.9.1
  Using cached urllib3-2.3.0-py3-none-any.whl (128 kB)
Collecting aiohttp>=3.0.0
  Using cached aiohttp-3.11.12-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (1.6 MB)
Collecting frozenlist>=1.1.1
```

Рисунок 2.5 - Установка библиотек.

Необходимо создать свой токен, для работы бота в телеграмме. Для этого находим определенного бота и регистрируем данные своего бота.

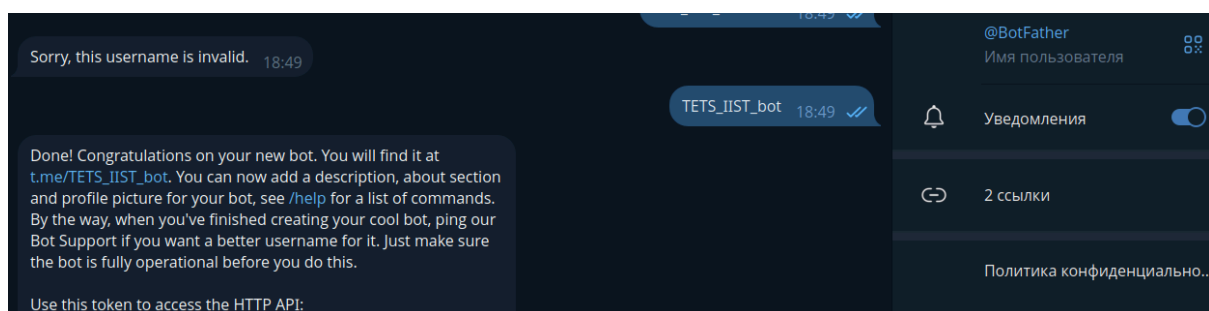
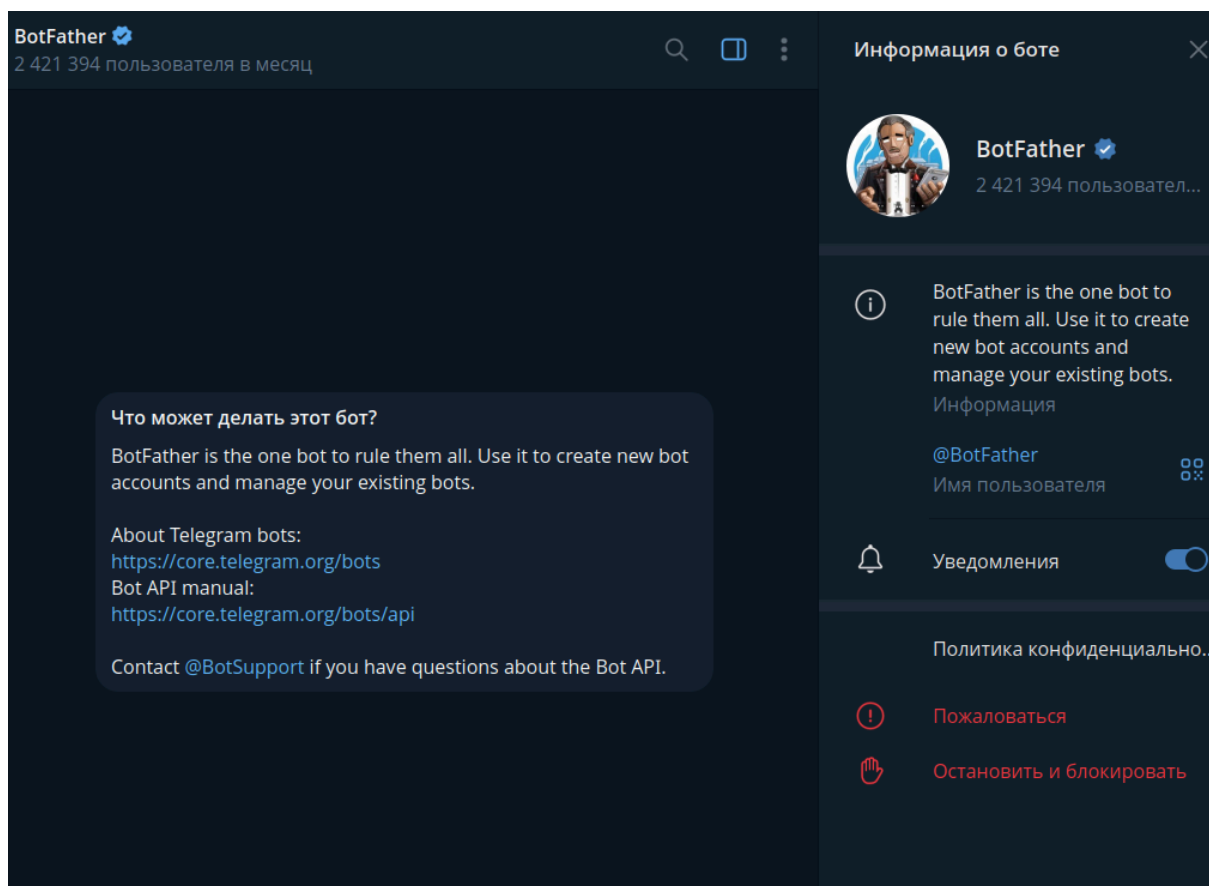


Рисунок 2.6 - 7 - Получение токена бота.

После получения токена, мы можем добавить скрипт работы нашего бота. и проверить его на работоспособность.

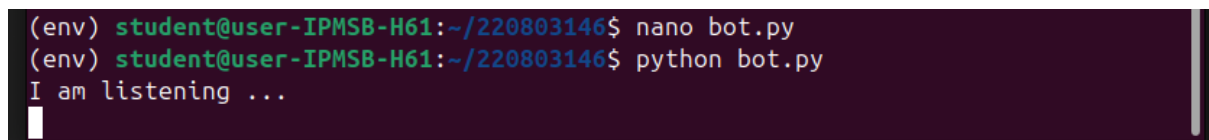


Рисунок 2.8 - Создание файла бота.

```
student@user-IPMSB-H61: ~/220803146
GNU nano 4.8 bot.py Modified
import telepot
import time
def handle(msg):
    chat_id = msg['chat']['id']
    command = msg['text']
    print('Got command: %s' % command)
    print('From : %s' % chat_id)
    if command == '/command1':
        bot.sendMessage(chat_id, 'Oks')
    elif command == '/command2':
        bot.sendMessage(chat_id, 'Ok')
bot = telepot.Bot('')
bot.message_loop(handle)
print('I am listening ...')
while 1:
    time.sleep(10)

^G Get Help      ^O Write Out    ^W Where Is     ^K Cut Text     ^J Justify      ^C Cur Pos
^X Exit          ^R Read File   ^\ Replace      ^U Paste Text   ^T To Spell     ^_ Go To Line
```

Рисунок 2.9 - Код бота.

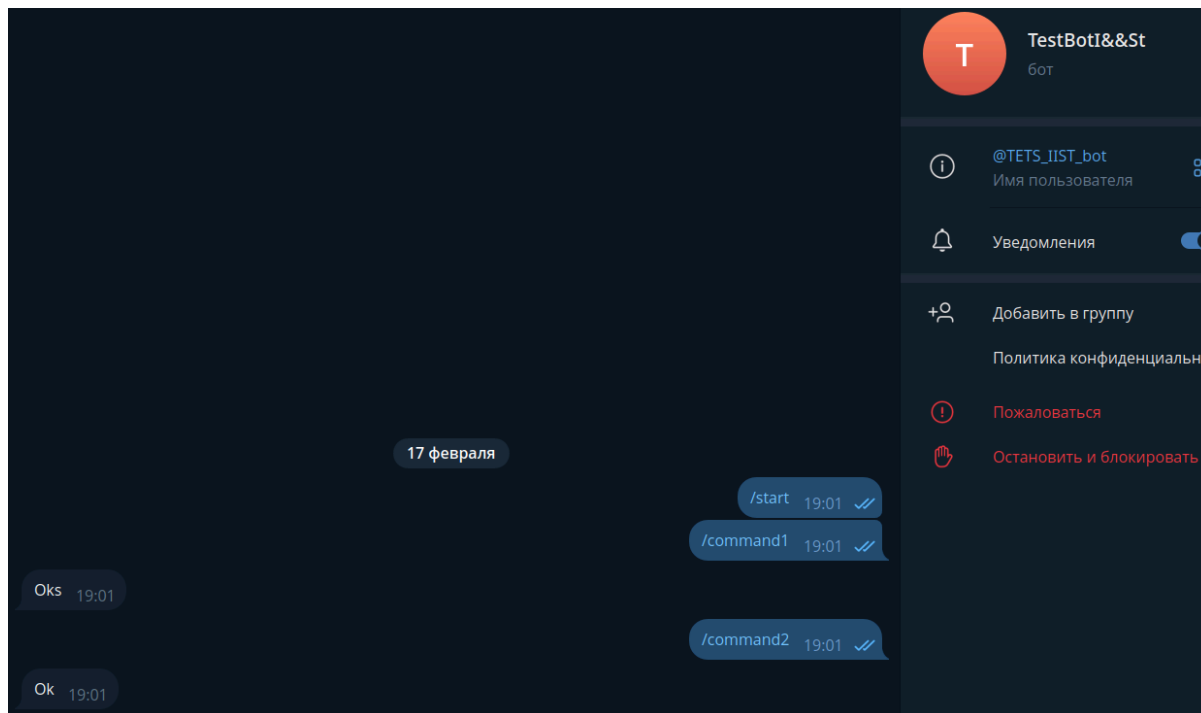
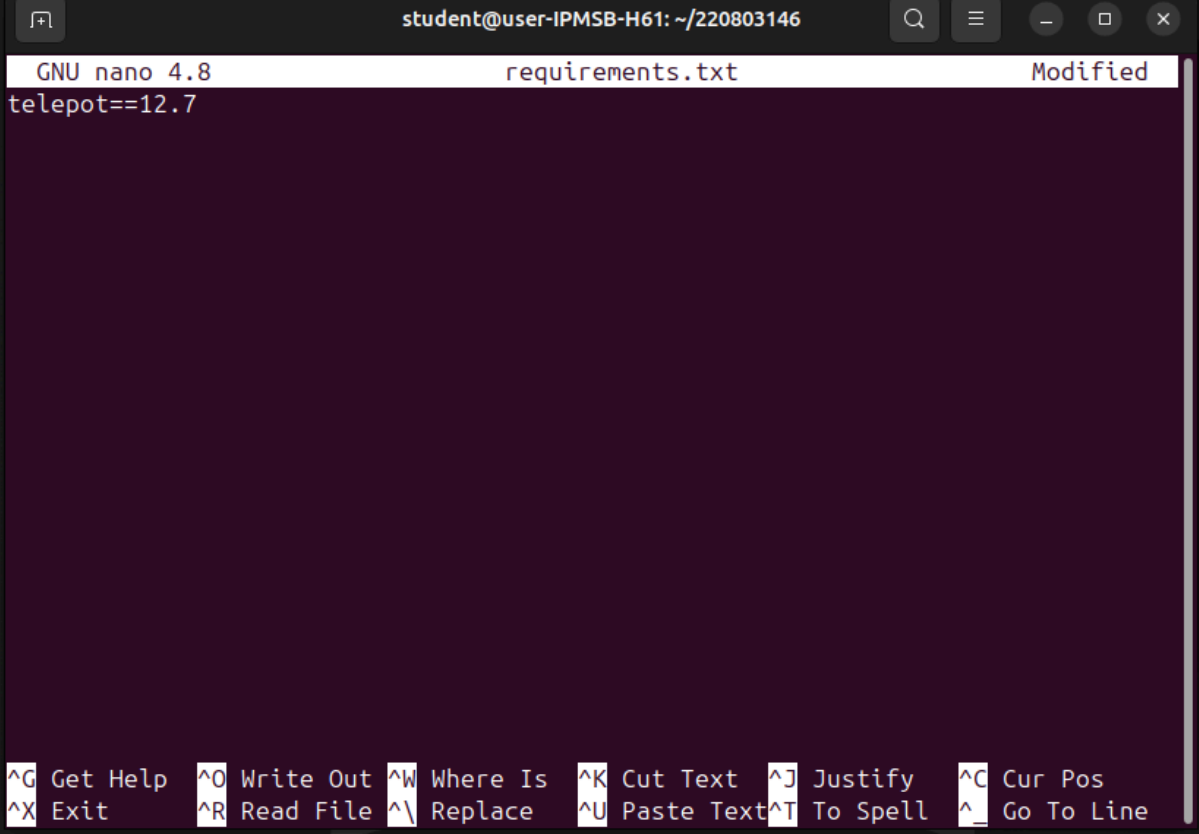


Рисунок 2.10 - Проверка работы бота.

Отлично, все работает. Далее необходимо создать docker-контейнер нашего бота. Но сначала необходимо создать файлы, которые доккер будет использовать для создания контейнера, это описание зависимостей (библиотек) и файл контейнера с данными о версии python и о том как запустить проект.



The image shows a terminal window with a dark background. At the top, a status bar indicates the user is 'student@user-IPMSB-H61: ~/220803146'. Below this, the nano text editor is open, editing a file named 'requirements.txt'. The editor's title bar shows 'GNU nano 4.8' and 'Modified'. The content of the file is 'telepot==12.7'. At the bottom of the terminal, a list of nano editor shortcuts is displayed, including ^G Get Help, ^O Write Out, ^W Where Is, ^K Cut Text, ^J Justify, ^C Cur Pos, ^X Exit, ^R Read File, ^\ Replace, ^U Paste Text, ^T To Spell, and ^\_ Go To Line.

```
student@user-IPMSB-H61: ~/220803146
GNU nano 4.8 requirements.txt Modified
telepot==12.7

^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify   ^C Cur Pos
^X Exit      ^R Read File ^\ Replace   ^U Paste Text ^T To Spell  ^_ Go To Line
```

Рисунок 2.11 - Файл requirements.txt.

```
student@user-IPMSB-H61: ~/220803146
GNU nano 4.8 Dockerfile Modified
FROM python:3.10 AS builder
COPY requirements.txt .
RUN pip install --user -r requirements.txt
FROM python:3.10-slim
WORKDIR /code
COPY --from=builder /root/.local /root/.local
COPY ./bot.py .
ENV PATH=/root/.local:$PATH
CMD [ "python", "-u", "./bot.py" ]

^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify   ^C Cur Pos
^X Exit      ^R Read File ^\ Replace   ^U Paste Text ^T To Spell  ^_ Go To Line
```

Рисунок 2.12 - Файл Dockerfile.

Далее создаем контейнер. Также стоит запустить контейнер чтобы проверить его работу.

```
student@user-IPMSB-H61:~/220803146$ docker build -t 220803146 .
[+] Building 9.2s (13/13) FINISHED docker:default
=> [internal] load build definition from Dockerfile 0.7s
=> => transferring dockerfile: 295B 0.0s
=> [internal] load metadata for docker.io/library/python:3.10-slim 2.9s
=> [internal] load metadata for docker.io/library/python:3.10 2.8s
=> [internal] load .dockerignore 0.7s
=> => transferring context: 2B 0.0s
=> [builder 1/3] FROM docker.io/library/python:3.10@sha256:76f22e4ce5377 0.0s
=> [stage-1 1/4] FROM docker.io/library/python:3.10-slim@sha256:66aad90b 0.0s
=> [internal] load build context 0.6s
=> => transferring context: 531B 0.0s
=> CACHED [stage-1 2/4] WORKDIR /code 0.0s
=> CACHED [builder 2/3] COPY requirements.txt . 0.0s
=> CACHED [builder 3/3] RUN pip install --user -r requirements.txt 0.0s
=> CACHED [stage-1 3/4] COPY --from=builder /root/.local /root/.local 0.0s
=> [stage-1 4/4] COPY ./bot.py . 1.2s
=> exporting to image 1.3s
=> => exporting layers 0.8s
=> => writing image sha256:a412c3fc63e70b7285f11235229f4348a503e38d61d66 0.0s
=> => naming to docker.io/library/220803146 0.2s
student@user-IPMSB-H61:~/220803146$
```

Рисунок 2.13 - Создание контейнера.

```
student@user-IPMSB-H61:~/220803146$ docker run -d --restart=always 220803146
cc1d2a29da48ffb09c7d
student@user-IPMSB-H61:~/220803146$
```

Рисунок 2.14 - Работа контейнера и его ID.

Далее можно посмотреть лог нашего образа. Для этого необходимо прописать `docker logs` и ID нашего контейнера.

```
student@user-IPMSB-H61:~/220803146$ docker logs cc1d2a29da48
I am listening ...
student@user-IPMSB-H61:~/220803146$
```

Рисунок 2.15 - Лог образа.

После всех проверок, останавливаем работу контейнера `docker stop`, и сохраняем его в архив, командой `docker save -o <путь к архиву> <имя образа>`.

```
student@user-IPMSB-H61:~/220803146$ docker save -o ./docker_image_220803146.tar 220803146
student@user-IPMSB-H61:~/220803146$
```

Рисунок 2.16 - Сохранение архива образа.

Далее скачиваем образ с сервера, командой `scp student@195.133.13.56:/home/student/docker_image_000000.tar .`

```
democran@democran-Modern-15-B5M:~$ scp student@193.124.118.93:/home/student/docker_image_2208031
46.tar .
student@193.124.118.93's password:
docker_image_220803146.tar                               10% 15MB 2.2MB/s 00:56 ETA
```

Рисунок 2.17 - Скачивание образа с сервера.

Можно запустить файл локально для проверки образа.

```
democran@democran-Modern-15-B5M:~/Рабочий стол/ИСТ/2 лаба доккер$ sudo docker run a412c3fc
63e7
I am listening ...
Traceback (most recent call last):
  File "/root/.local/lib/python3.10/site-packages/urllib3/connectionpool.py", line 464, in
make_request
```

Рисунок 2.18 - Запуск образа локально.

**Вывод:** в данной практической работе, изучили работу с сервером через консоль, такие аспекты как подключение, работа в файловой системе, правильно работать без вреда для самого сервера. Разработали бота с использованием библиотеки `telepot`, научившись узнавать токен. Работа с `docker` и создание `docker`-контейнера для бота, включая подготовку файлов `requirements.txt` и `Dockerfile`, показало, что можно

удобно переносить свое приложение с любого сервера, что помогает в разработке и развертывании проектов на новом сервере.