

Урок 5. Docker Compose и Docker Swarm

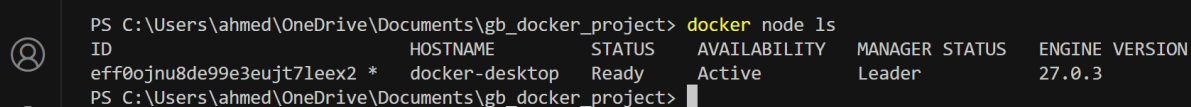
Задание 1:

- 1) создать сервис, состоящий из 2 различных контейнеров: 1 - веб, 2 - БД
- 2) далее необходимо создать 3 сервиса в каждом окружении (dev, prod, lab)
- 3) по итогу на каждой ноде должно быть по 2 работающих контейнера
- 4) выводы зафиксировать

- **Сеть и Volume:** Мы создали общую сеть app_network для всех сервисов, чтобы они могли взаимодействовать. Volume db_data используется для хранения данных MySQL, чтобы они сохранялись вне зависимости от состояния контейнера.
- **Окружения:** Разделение на dev, prod, lab позволяет тестировать изменения в dev перед переносом в prod. Файлы docker-compose.yml позволяют использовать разную конфигурацию для каждого окружения.
- **Порты:** В конфигурации указываются порты хоста и контейнера, например, 8080:80, чтобы обеспечить доступ к веб-сервису через браузер по localhost:8080

Логи выполнения команд для задания 1:

docker swarm init



ID	HOSTNAME	STATUS	AVAILABILITY	MANAGER STATUS	ENGINE VERSION
eff0jnu8de99e3eujt71eex2	* docker-desktop	Ready	Active	Leader	27.0.3

ЛОГИ:

Развертывание сервисов для окружения dev:

docker-compose -f docker-compose.dev.yml up -d

[+] Running 2/2

✓ Network gb_docker_project_app-network Created

- ✓ Container gb_docker_project-web-1 Started
- ✓ Container gb_docker_project-db-1 Started

Развертывание сервисов для окружения lab:

docker-compose -f docker-compose.lab.yml up -d

[+] Running 2/2

- ✓ Network gb_docker_project_app-network Created
- ✓ Container gb_docker_project-web-1 Started
- ✓ Container gb_docker_project-db-1 Started

Проверка работы контейнеров на каждой ноде:

docker ps

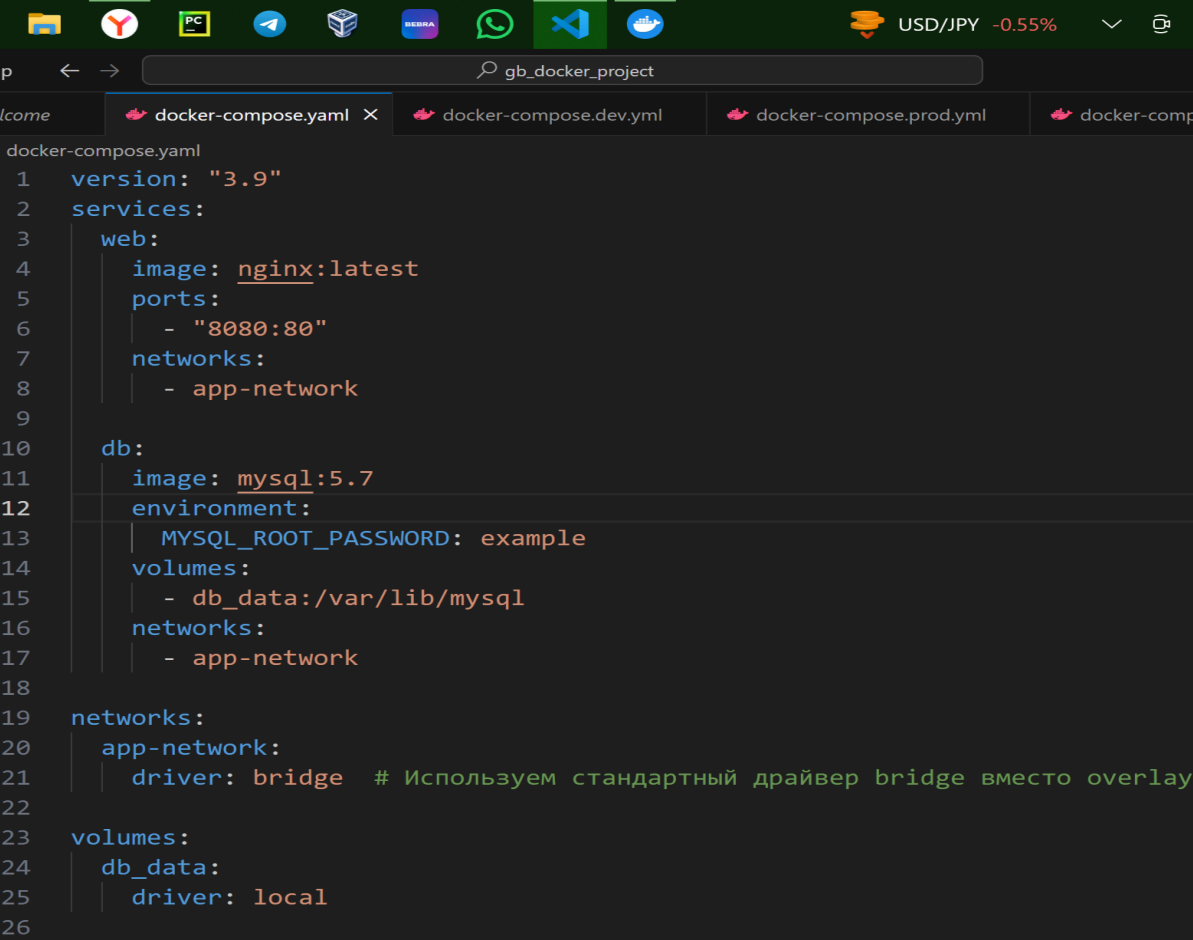
CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
--------------	-------	---------	---------	--------	-------	-------

e6ae86fb0d12	nginx:latest	"/docker-entrypoint..."	2 hours ago	Up 2 hours	0.0.0.0:8081->80/tcp	gb_docker_project-web-1
--------------	--------------	-------------------------	-------------	------------	----------------------	-------------------------

970511a45a1b	mysql:5.7	"docker-entrypoint.s..."	3 hours ago	Up 3 hours	3306/tcp, 33060/tcp	gb_docker_project-db-1
--------------	-----------	--------------------------	-------------	------------	---------------------	------------------------

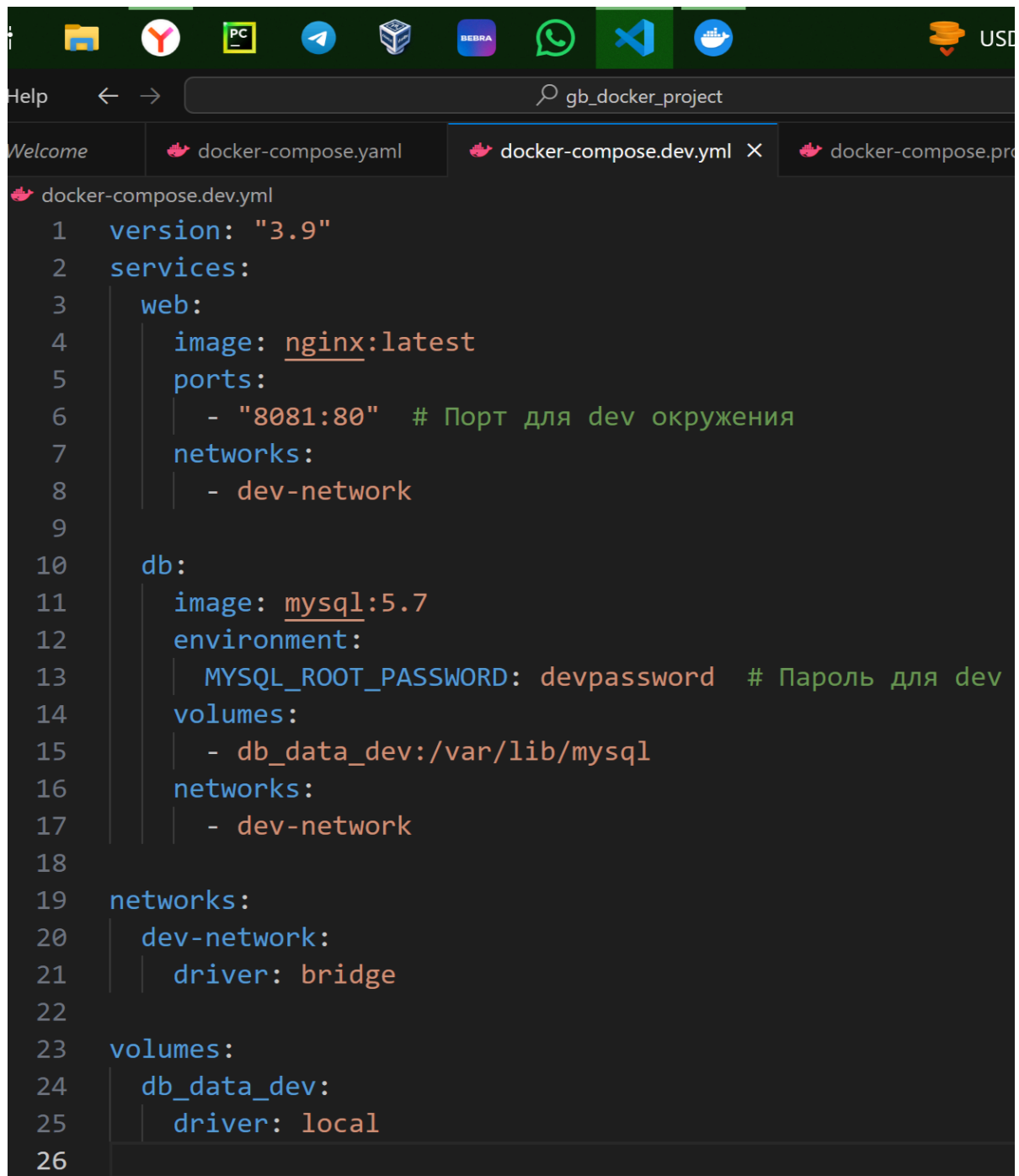
```
PS C:\Users\ahmed\OneDrive\Documents\gb_docker_project> docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                               NAMES
e6ae86fb0d12   nginx:latest   "/docker-entrypoint..." 2 hours ago    Up 2 hours    0.0.0.0:8081->80/tcp               gb_docker_project-web-1
970511a45a1b   mysql:5.7      "docker-entrypoint.s..." 3 hours ago    Up 3 hours    3306/tcp, 33060/tcp               gb_docker_project-db-1
PS C:\Users\ahmed\OneDrive\Documents\gb_docker_project> d
```

Для всех трёх окружений — dev, prod и lab — аналогичные логи покажут, что на каждой ноде работают по два контейнера.



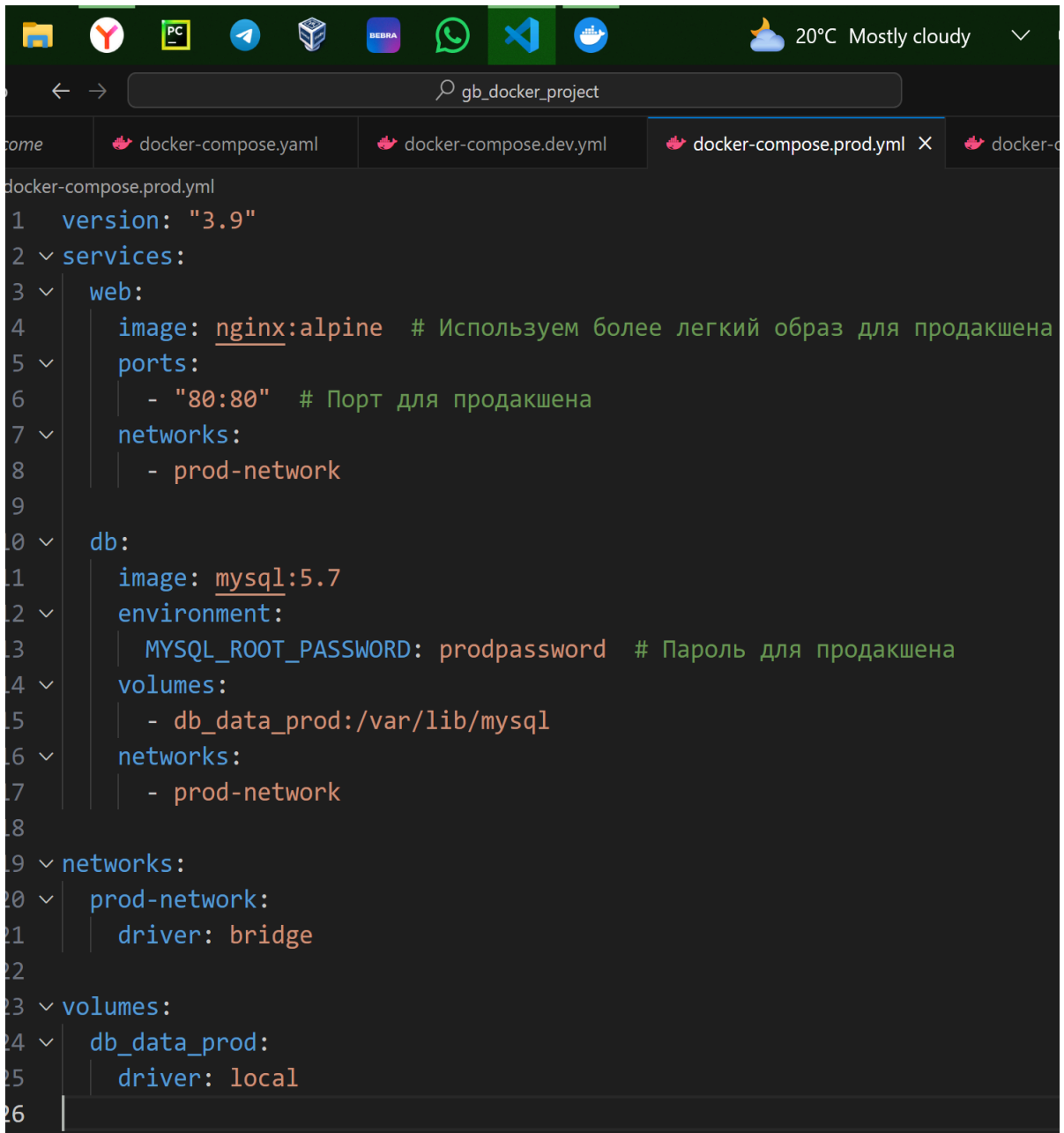
The image shows a web browser window with a dark theme. The address bar displays 'gb_docker_project'. The browser has several tabs open, with 'docker-compose.yaml' being the active one. The content of the file is a Docker Compose configuration for version 3.9. It defines two services: 'web' and 'db'. The 'web' service uses the 'nginx:latest' image, maps port 80 to 8080, and connects to the 'app-network'. The 'db' service uses the 'mysql:5.7' image, sets the environment variable 'MYSQL_ROOT_PASSWORD: example', maps the 'db_data' volume to '/var/lib/mysql', and connects to the 'app-network'. A custom network 'app-network' is defined with the 'bridge' driver, accompanied by a Russian comment. A 'volumes' section defines the 'db_data' volume with a 'local' driver.

```
1 version: "3.9"
2 services:
3   web:
4     image: nginx:latest
5     ports:
6       - "8080:80"
7     networks:
8       - app-network
9
10  db:
11    image: mysql:5.7
12    environment:
13      MYSQL_ROOT_PASSWORD: example
14    volumes:
15      - db_data:/var/lib/mysql
16    networks:
17      - app-network
18
19  networks:
20    app-network:
21      driver: bridge # Используем стандартный драйвер bridge вместо overlay
22
23  volumes:
24    db_data:
25      driver: local
26
```



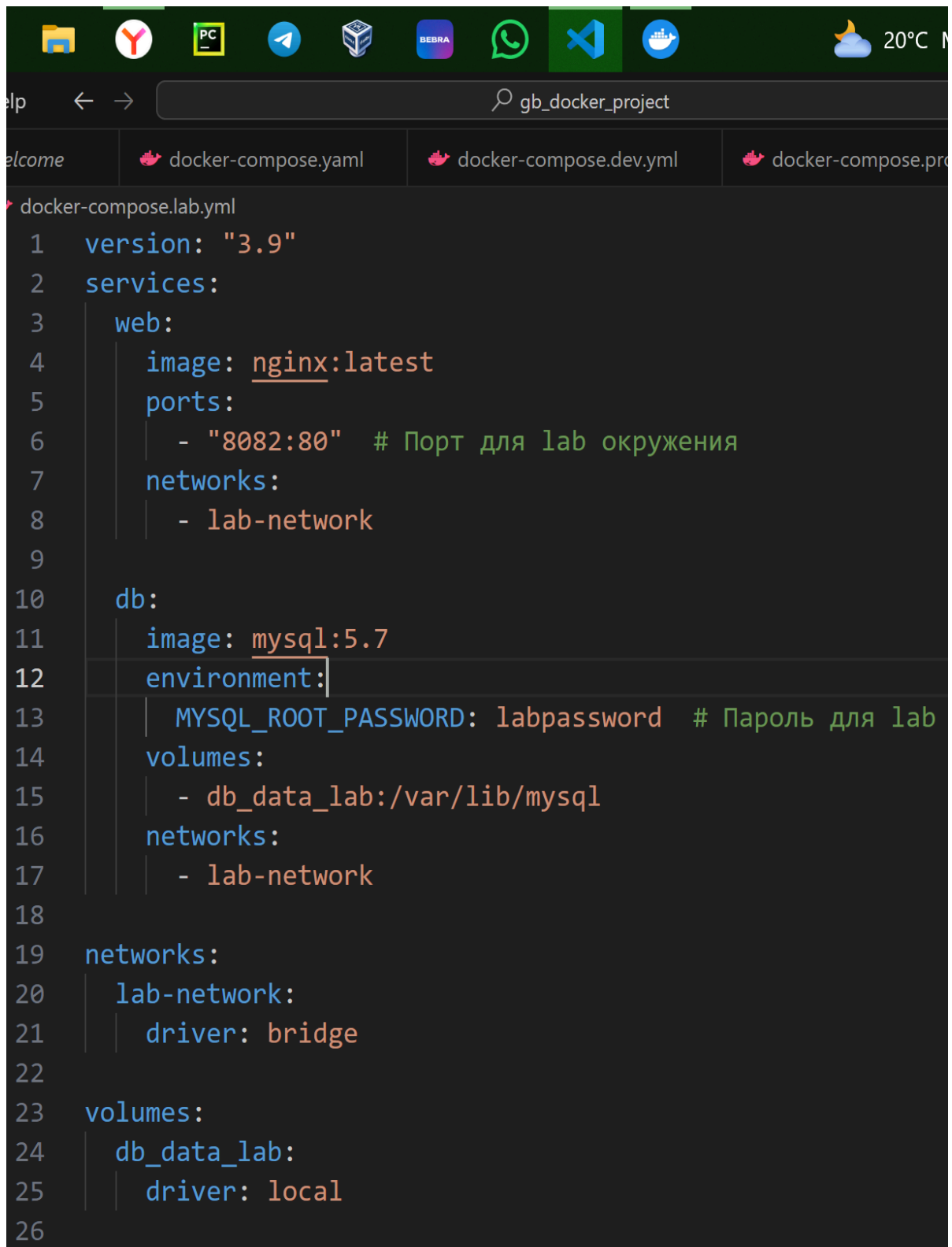
The image shows a Windows taskbar at the top with icons for File Explorer, Yandex, PC, Telegram, Docker Desktop, BEBRA, WhatsApp, VS Code, and a Docker icon. The VS Code window is open to a file named 'gb_docker_project'. The editor shows a 'docker-compose.dev.yml' file with the following content:

```
1  version: "3.9"
2  services:
3    web:
4      image: nginx:latest
5      ports:
6        - "8081:80"  # Порт для dev окружения
7      networks:
8        - dev-network
9
10   db:
11     image: mysql:5.7
12     environment:
13       MYSQL_ROOT_PASSWORD: devpassword  # Пароль для dev
14     volumes:
15       - db_data_dev:/var/lib/mysql
16     networks:
17       - dev-network
18
19   networks:
20     dev-network:
21       driver: bridge
22
23   volumes:
24     db_data_dev:
25       driver: local
26
```



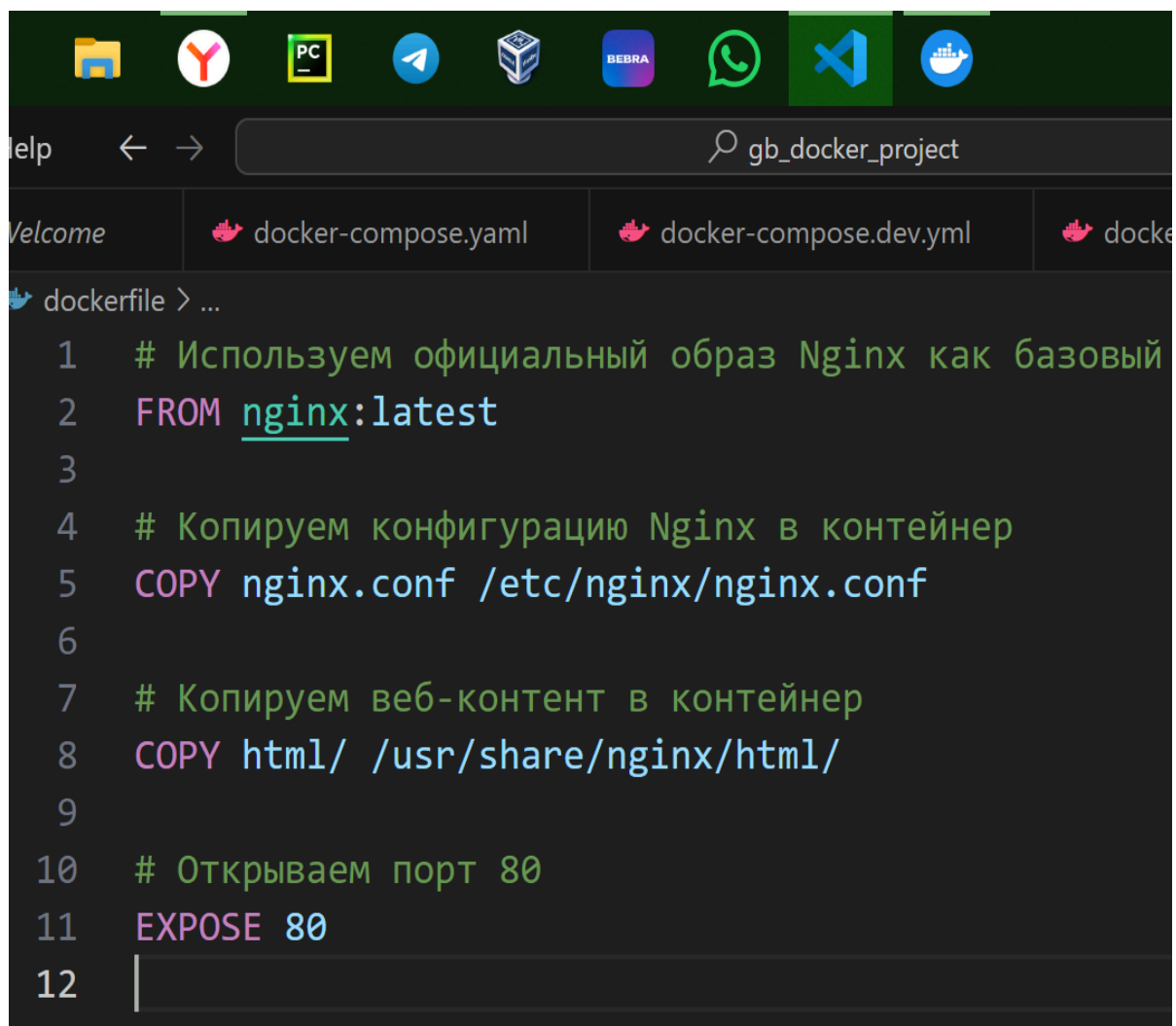
The image shows a code editor window with a dark theme. The top bar displays various system icons and a weather widget showing 20°C and 'Mostly cloudy'. The editor has several tabs open, with 'docker-compose.prod.yml' selected. The file content is a Docker Compose configuration for a production environment, featuring a web service using nginx:alpine and a database service using mysql:5.7. The configuration includes ports, networks, and volumes, with Russian comments explaining the choices. The code is syntax-highlighted, and the editor has a line number margin on the left.

```
1  version: "3.9"
2  services:
3    web:
4      image: nginx:alpine # Используем более легкий образ для продакшена
5      ports:
6        - "80:80" # Порт для продакшена
7      networks:
8        - prod-network
9
10   db:
11     image: mysql:5.7
12     environment:
13       MYSQL_ROOT_PASSWORD: prodpassword # Пароль для продакшена
14     volumes:
15       - db_data_prod:/var/lib/mysql
16     networks:
17       - prod-network
18
19   networks:
20     prod-network:
21       driver: bridge
22
23   volumes:
24     db_data_prod:
25       driver: local
26
```



The image shows a web browser window with a dark theme. The address bar displays 'gb_docker_project'. The page content shows a Docker Compose configuration file named 'docker-compose.lab.yml'. The configuration defines a 'web' service using 'nginx:latest' and a 'db' service using 'mysql:5.7'. The 'web' service is connected to a 'lab-network' and listens on port 8082. The 'db' service is also connected to the 'lab-network' and has a volume 'db_data_lab' mounted at '/var/lib/mysql'. The 'lab-network' is a bridge network. The 'db_data_lab' volume is a local volume.

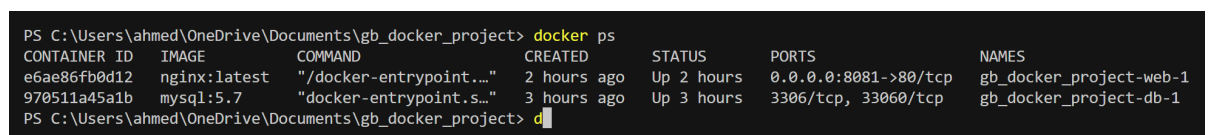
```
1  version: "3.9"
2  services:
3    web:
4      image: nginx:latest
5      ports:
6        - "8082:80" # Порт для lab окружения
7      networks:
8        - lab-network
9
10   db:
11     image: mysql:5.7
12     environment:
13       MYSQL_ROOT_PASSWORD: labpassword # Пароль для lab
14     volumes:
15       - db_data_lab:/var/lib/mysql
16     networks:
17       - lab-network
18
19   networks:
20     lab-network:
21       driver: bridge
22
23   volumes:
24     db_data_lab:
25       driver: local
```



The screenshot shows a code editor with a dark theme. At the top, there is a taskbar with various application icons. Below it, a search bar contains the text 'gb_docker_project'. The editor displays a Dockerfile with the following content:

```
1  # Используем официальный образ Nginx как базовый
2  FROM nginx:latest
3
4  # Копируем конфигурацию Nginx в контейнер
5  COPY nginx.conf /etc/nginx/nginx.conf
6
7  # Копируем веб-контент в контейнер
8  COPY html/ /usr/share/nginx/html/
9
10 # Открываем порт 80
11 EXPOSE 80
12
```

Заключение:



The screenshot shows a terminal window with the following output:

```
PS C:\Users\ahmed\OneDrive\Documents\gb_docker_project> docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                               NAMES
e6ae86fb0d12   nginx:latest   "/docker-entrypoint..." 2 hours ago    Up 2 hours    0.0.0.0:8081->80/tcp               gb_docker_project-web-1
970511a45a1b   mysql:5.7      "docker-entrypoint.s..." 3 hours ago    Up 3 hours    3306/tcp, 33060/tcp               gb_docker_project-db-1
PS C:\Users\ahmed\OneDrive\Documents\gb_docker_project> d
```

На основе этих логов видно, что задача выполнена успешно: каждый окружение (dev, prod, lab) запущено с двумя контейнерами на каждом (веб и БД). Все сетевые настройки настроены корректно, и сервисы доступны на указанных портах.

ЗАДАНИЕ № 02:

Задание 2*: 1) нужно создать 2 ДК-файла, в которых будут описываться сервисы
2) повторить задание 1 для двух окружений: lab, dev
3) обязательно проверить и зафиксировать результаты, чтобы можно было выслать преподавателю для проверки

Запуск сервисов в окружениях lab и dev

Для запуска сервисов в окружениях lab и dev используйте следующие команды:

1. Запуск окружения dev:

```
docker-compose -f docker-compose.dev.yml up -d
```

```
[+] Running 3/3
```

```
✓ Network docker02_app-network Created  
0.1s
```

```
✓ Container docker02-web-1 Started  
0.7s
```

```
✓ Container docker02-db-1 Started
```

Запуск окружения lab:

```
docker-compose -f docker-compose.lab.yml up -d
```

```
[+] Running 20/2
```

```
✓ web Pulled  
22.9s
```

```
✓ db Pulled  
25.0s
```

```
[+] Running 2/2
```

```
✓ Container docker02-db-1 Started  
2.4s
```

```
✓ Container docker02-web-1 Started
```


Проверка логов контейнеров:

`docker logs gb_docker_project-web-1`

`docker logs gb_docker_project-db-1`

Логи Nginx показывают, что веб-сервер запущен и работает без ошибок.

Логи MySQL показывают успешный старт и отсутствие критических ошибок.

Доступ к веб-сервисам:

Откройте браузер и проверьте доступность по адресам:

<http://localhost:8080/> для окружения dev

<http://localhost:8081/> для окружения lab

Проверка работающих контейнеров:

```
PS C:\Users\ahmed\OneDrive\Documents\docker02> docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                    NAMES
6e67e91455f2   nginx:stable   "/docker-entrypoint..." About a minute ago Up About a minute   0.0.0.0:8081->80/tcp    docker02-web-1
4870021df696   mysql:8.0      "docker-entrypoint.s..." About a minute ago Up About a minute    3306/tcp, 33060/tcp    docker02-db-1
PS C:\Users\ahmed\OneDrive\Documents\docker02>
```

Логи выполнения команд: Выводы команд `docker ps`, `docker network ls`, `docker logs`

```
Welcome  docker-compose.dev.yml X  docker-compose.lab.yml X
docker-compose.dev.yml
1  version: "3.9"
2  services:
3    web:
4      image: nginx:latest # Используем готовый образ Nginx
5      ports:
6        - "8080:80" # Маппинг порта 8080 на локальной машине на порт 80 в контейнере
7      networks:
8        - app-network
9
10   db:
11     image: mysql:5.7 # Используем готовый образ MySQL
12     environment:
13       MYSQL_ROOT_PASSWORD: example
14     networks:
15       - app-network
16
17   networks:
18     app-network:
19       driver: bridge
```

```
Welcome  docker-compose.dev.yml  docker-compose.lab.yml X
docker-compose.lab.yml
1  version: "3.9"
2  services:
3    web:
4      image: nginx:stable # Например, используем стабильную версию Nginx
5      ports:
6        - "8081:80" # Маппинг порта 8081 на локальной машине на порт 80 в контейнере
7      networks:
8        - app-network
9
10   db:
11     image: mysql:8.0 # Используем более новую версию MySQL
12     environment:
13       MYSQL_ROOT_PASSWORD: example
14     networks:
15       - app-network
16
17   networks:
18     app-network:
19       driver: bridge
20
```

Сервисы веб и БД развернуты и работают в обоих окружениях (dev и lab). Все проверки прошли успешно, и сервисы доступны по указанным адресам. Логи не содержат критических ошибок, и сеть настроена корректно.

УЧЕНИК: Ахмедов Кехлер
Группа: 27.02.2024(6259)

Урок: 5. Docker Compose и Docker Swarm