Если вы видите что-то необычное, просто сообщите мне.

Подготовка сервера для обновления версии приложения.

В этой инструкции мы попытаемся объединить уже ранее собранные нами конфигурационные файлы во едино. Что для этого нужно

Подготовим север для запуска сервиса. Установим необходимое ПО

```
#!/bin/bash
apt update
apt install -y docker.io nginx
groupadd docker
usermod -aG docker $USER
sudo curl -L "https://github.com/docker/compose/releases/download/1.29.2/docker-compose-
$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
 sudo chmod +x /usr/local/bin/docker-compose
 sudo ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose
```

Настроим docker-compose и запустим проект

Настройку проекта мы можем опустить, там как мы её уже изучали в разделе про docker-compose. Обычно к этому моменту мы должны уже иметь репозиторий, который должен работать локально. Его достаточно склонировать, и использовать для запуска.

В проекте не должны использоваться порты 80 и 443, так как это порты используемые nginx.

Для полноценной работы, нам необхдимо изменить адрес контейнера в docker-compose, так как теперь мы не будем собирать контейнер на сервер, за нас это делает pipeline, нам необходимо заменить директиву `build` на `image` таким образом, мы скажем докеру, что ему необходимо брать образ откуда то еще.

Настроим скрипт выливки на сервер и проверим его работоспособность.

Добавим в проект `deploy.sh` - скрипт который позволит нам актоматически равзертывать новую версию на сайте.

```
#!/usr/bin/env bash



#Не забываем авторизоваться в docker, если используем закрытый или частный репозиторий
docker login -u КАКОЙ_ТО-token -p КАКОЙ-ТО-ПАРОЛЬ какой-то.регистри.com

cd "АДРЕС/МЕСТОПОЛОЖЕНИЯ/ПАПКИ/ПРОЕКТА"
```

```
# Вытягиваем новую версию
docker-compose pull;
# Останавливаем старый сервис
docker-compose down;
# Запускаем новую версию
docker-compose up -d
```

Настраиваем nginx и получаем рабочий сервис.

Теперь мы дошли до того, чтобы настроить nginx. Nginx нужен для того, чтобы мы могли настроить работу сервиса по доменному имени.

Для работы нашего сервиса воспользуемся готовым конфигом nginx.conf

```
server{
       listen 80;
       server_name videomanager-test.garpix.com;

       location / {
            proxy_pass http://web:8080;
            proxy_set_header Host $http_host;
            proxy_set_header Connection "upgrade";
            proxy_set_header X-Real-IP $remote_addr;
            proxy_set_header X-Forwarded-Proto $scheme;
            proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
            proxy_set_header Upgrade $http_upgrade;
            proxy_connect_timeout       600;
            proxy_send_timeout          600;
            proxy_read_timeout          600;
            send_timeout                600;
        }
}
```

Revision #4
Created 2021-09-27 15:14:07 UTC by gasick

Updated 2023-11-08 16:10:47 UTC by gasick