**AKKUYU NÜKLEER ANONİM ŞİRKETİ**

|  |  |
| --- | --- |
|  | УТВЕРЖДЕНО  Заместитель технического директора по ремонту |
|  | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Шульгин С.В.  «\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2024 г. |

Руководство системного администратора

Информационной системы «Временное решение электронный журнал дефектов»

|  |  |
| --- | --- |
| СОГЛАСОВАНО  Начальник ЦИКТ  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Постников Д.А.  «\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2024 г. |  |
|  |  |
|  |  |

ЛИСТ ИЗМЕНЕНИЙ

| Дата версии | Номер версии | Краткое описание изменений | Ответственный |
| --- | --- | --- | --- |
|  | 1.0 | Документ создан |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

ЛИСТ СОГЛАСОВАНИЙ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Роль на проекте | Должность | Фамилия Имя  Отчество | Дата | Подпись |
| Функциональный заказчик ИТ - мероприятия | Начальник ОУР | Лебедев И.И. |  |  |
| Руководитель ИТ - мероприятия | Главный специалист ОУР | Кудряшов Н.И. |  |  |
| Главный разработчик ИТ - мероприятия | Главный специалист ЦИКТ | Федоров А.В. |  |  |
| Администратор и аналитик ИТ - мероприятия | Начальник участка ЦИКТ | Егошина В.Л. |  |  |
| Разработчик документа | Специалист-программист 1 категории | Петров Т.С. |  |  |

Содержание

[ЛИСТ ИЗМЕНЕНИЙ 3](#_Toc167179580)

[ЛИСТ СОГЛАСОВАНИЙ 4](#_Toc167179581)

[Термины и Сокращения 7](#_Toc167179582)

[Перечень используемых сокращений 7](#_Toc167179583)

[Перечень используемых терминов 7](#_Toc167179584)

[**Назначение** 9](#_Toc167179585)

[**1. Состав ВР «ЭЖД»** 10](#_Toc167179586)

[**1.1. Серверы и их назначение** 10](#_Toc167179587)

[**1.2. Компоненты ВР «ЭЖД»** 10](#_Toc167179588)

[**1.3. Организация сети между контейнерами ВР «ЭЖД»** 11](#_Toc167179589)

[**2. Обновление компонентов ВР «ЭЖД»** 13](#_Toc167179590)

[**2.1. Обновление приложения с использованием GUI** 13](#_Toc167179591)

[**2.2. Обновление приложения с использованием CLI** 24](#_Toc167179592)

[**3. Организация резервного копирования** 26](#_Toc167179593)

[**3.1. Автоматическое резервное копирование БД** 26](#_Toc167179594)

[**3.2. Ручное резервное копирование БД** 28](#_Toc167179595)

[**3.2.1. Резервное копирование БД с использованием pgAdmin** 28](#_Toc167179596)

[**3.2.2. Резервное копирование БД с использованием pg\_dump** 31](#_Toc167179597)

[**4. Восстановление БД из резервной копии** 32](#_Toc167179598)

[**4.1. Восстановление БД с использованием pgAdmin** 32](#_Toc167179599)

[**4.2. Восстановление БД с использованием pg\_restore** 34](#_Toc167179600)

[**5. Хранение резервных копий** 35](#_Toc167179601)

[**5.1. Хранение резервных копий БД, созданных автоматически** 35](#_Toc167179602)

[**5.2. Хранение резервных копий БД, созданных вручную** 35](#_Toc167179603)

[**5.3. Хранение резервных копий контейнеров и образов ВР «ЭЖД», созданных вручную** 35](#_Toc167179604)

[**6.1. Ошибка при авторизации пользователя в ВР «ЭЖД»** 36](#_Toc167179605)

[**6.2. Запуск новых контейнеров через CLI в случае ошибок на текущих контейнерах в ВР «ЭЖД»** 38](#_Toc167179606)

[**6.2.1. Создание нового контейнера defect\_app** 38](#_Toc167179607)

[**6.2.2. Создание нового контейнера postgres-conteiner** 38](#_Toc167179608)

[**6.2.3. Создание нового контейнера pgadmin4** 39](#_Toc167179609)

[**6.2.4. Создание нового контейнера redis\_7\_2\_4\_cont** 40](#_Toc167179610)

# Термины и Сокращения

## Перечень используемых сокращений

| Сокращение | Значение |
| --- | --- |
| ВР «ЭЖД» | Временное решение «Электронный журнал дефектов» |
| Компания | AKKUYU NÜKLEER ANONİM ŞİRKETİ |
| MS AD | Microsoft Active Directory |
| CLI | Command Line Interface |
| TCP | Transmission Control Protocol |
| GUI | Graphical User Interface |
| SSH | Secure Shell |
| IP | Internet protocol |
| БД | База данных |
| ОС | Операционная система |
| ПО | Программное обеспечение |
| СУБД | Система управления базами данных |
| ЦИКТ | Цех информационно-коммуникационных технологий |
| УИТ | Управление информационных технологий |

## Перечень используемых терминов

В данном документе применяются следующие термины:

**Системный администратор** – специалист Компании, который занимается обеспечением надежной и безопасной работы ВР «ЭЖД», осуществляет обновление и поддержку системы.

**Docker контейнер** – стандартизированный пакет программного обеспечения, который включает в себя приложение и все его зависимости, такие как библиотеки, исполняемые файлы и другие файлы, необходимые для его выполнения. Контейнеры изолированы друг от друга и от окружающей системы, что делает их переносимыми и обеспечивает однородность в различных средах выполнения.

**Образ Docker** – шаблон, содержащий все необходимые инструкции и зависимости для создания и запуска конкретного контейнера. Он включает в себя файлы системы, среду выполнения и настройки, необходимые для работы приложения в изолированном окружении.

**Docker Hub** – облачный реестр, предоставляемый Docker, который позволяет пользователям находить, загружать и распространять контейнерные образы.

**Dockerfile** – текстовый файл, содержащий инструкции для создания Docker-образа. Он определяет окружение и настройки контейнера, включая базовый образ, установку программного обеспечения, настройку переменных среды, копирование файлов и многое другое. Dockerfile используется для автоматизации процесса создания образов, что делает его воспроизводимым и масштабируемым.

**CLI** – интерфейс командной строки. Это способ взаимодействия с компьютерной программой или операционной системой путем ввода текстовых команд в командной строке или терминале. CLI обычно используется для выполнения задач, автоматизации операций и управления компьютерными ресурсами.

**GUI** – графический пользовательский интерфейс. Это способ взаимодействия с компьютерной программой или операционной системой с помощью графических элементов, таких как окна, кнопки, меню и диалоговые окна. GUI облегчает визуализацию и выполнение задач пользователя с помощью манипуляции объектами на экране с помощью мыши, клавиатуры или сенсорного экрана.

**Резервное копирование** – процесс создания копии данных с целью сохранения их в случае потери, повреждения или случайного удаления. Резервные копии могут быть созданы на физических носителях, таких как жесткие диски, или в облачном хранилище. Этот процесс является важной частью стратегии защиты данных и обеспечивает возможность восстановления информации в случае чрезвычайных ситуаций.

**PostgreSQL** – система управления реляционными базами данных с открытым исходным кодом.

**pgAdmin** – бесплатное программное обеспечение с открытым исходным кодом, предназначенное для администрирования и управления серверами баз данных PostgreSQL. Оно предоставляет графический интерфейс пользователя для выполнения различных задач.

**pg\_dump** – утилита командной строки, предоставляемая PostgreSQL, которая используется для создания резервных копий баз данных PostgreSQL.

**pg\_restore** – утилита командной строки для восстановления баз данных PostgreSQL.

**Microsoft Active Directory** – служба каталогов и служба аутентификации, разработанная Microsoft, которая используется в среде Windows для управления сетевыми ресурсами и предоставления единой точки аутентификации и авторизации для пользователей, компьютеров и других объектов в сети.

**Тестовое окружение приложения** – это специально настроенное окружение, в котором разработчики и тестировщики проверяют работоспособность и корректность приложения перед его выпуском.

**Продакшн-окружение приложения** – это рабочее окружение, в котором приложение доступно конечным пользователям. В этом окружении приложение обрабатывает реальные данные и выполняет свои основные функции.

**NoSQL** – это тип баз данных, которые не используют реляционную модель и SQL для управления данными. NoSQL базы данных предназначены для хранения и обработки больших объемов разнородных данных и часто используются в распределенных системах.

**URL (Uniform Resource Locator)** – это унифицированный указатель ресурса, который используется для идентификации и доступа к ресурсам в Интернете.

**PuTTY** – свободно распространяемый клиент для удаленного доступа к компьютерам по протоколам SSH, Telnet, Rlogin и серийному порту. Он позволяет пользователям устанавливать безопасное соединение с удаленными системами и осуществлять управление ими через текстовый интерфейс.

**Cron** – стандартная система в Unix-подобных операционных системах для планирования выполнения задач в определенное время или периодически.

**Псевдотерминал** – программный интерфейс, который предоставляет терминальные возможности для процессов, запущенных в виртуальном окружении.

**Bash-оболочка (Bash shell)** – интерпретатор командной строки и сценариев для Unix-подобных операционных систем, таких как Linux.

**Назначение**

Объектом обслуживания по данному документу является временное решение «Электронный журнал дефектов» (далее по тексту – ВР «ЭЖД»).

ВР «ЭЖД» предназначена для обеспечения наличия работоспособной системы управления дефектами оборудования до начала подэтапа А-3, накопления базы дефектов по оборудованию до ввода в эксплуатацию ЦС ТОиР.

**1. Состав ВР «ЭЖД»**

**1.1. Серверы и их назначение**

Серверы, обеспечивающие работу ВР «ЭЖД»:

* **Основной сервер**:

Используется для запуска в работу контейнеров с приложением и его компонентами.

* IP-адрес: <ip\_адрес\_основного\_сервера>
* Адрес входа в приложение: https://<URL\_для\_входа\_в\_приложение>/
* ОС: Ubuntu Server
* ПО: Docker - запуск приложения и его компонентов в контейнерах; Docker compose - средство создания многоконтейнерных приложений
* **Резервный сервер**:

Используется для запуска скрипта бэкапа БД по расписанию и хранения резервных копий БД.

* IP-адрес: <ip\_адрес\_резервного\_сервера>
* ОС: Ubuntu Server
* ПО: Docker - запуск приложения и его компонентов в контейнерах; Docker compose - средство создания многоконтейнерных приложений

**1.2. Компоненты ВР «ЭЖД»**

На основном сервере развернуты следующие Docker контейнеры:

* + - **defect\_test**: контейнер с приложением ВР «ЭЖД» в рамках тестового окружения. Доступен на порту 4000 протокола TCP;
    - **defect\_app**: контейнер с приложением ВР «ЭЖД» в рамках продакшн-окружения. Доступен на порту 443 протокола TCP;
    - **redirect\_http\_to\_https**: контейнер для перенаправления с HTTP на HTTPS адреса приложения. Доступен на порту 80 протокола TCP;
    - **pgadmin4**: контейнер для администрирования и управления серверами баз данных PostgreSQL с графическим клиентом. Доступен на порту 9000 протокола TCP;
    - **redis\_7\_2\_4\_cont**: контейнер с СУБД класса NoSQL для кэширования данных. Прослушивает порт 6379 протокола TCP;
    - **portainer**: контейнер с инструментом для управления Docker контейнерами. Доступен на портах 9443 и 8000 протокола TCP;
    - **postgres-conteiner**: контейнер с СУБД PostgreSQL. Прослушивает порт 5432 протокола TCP;
    - **nginx\_zaglushka\_1**: контейнер для отображения сервисного сообщения о технических работах при переходе по адресу входа в приложение. Включается вручную, если ведутся технические работы на сервере. Использует порт 443 протокола TCP;

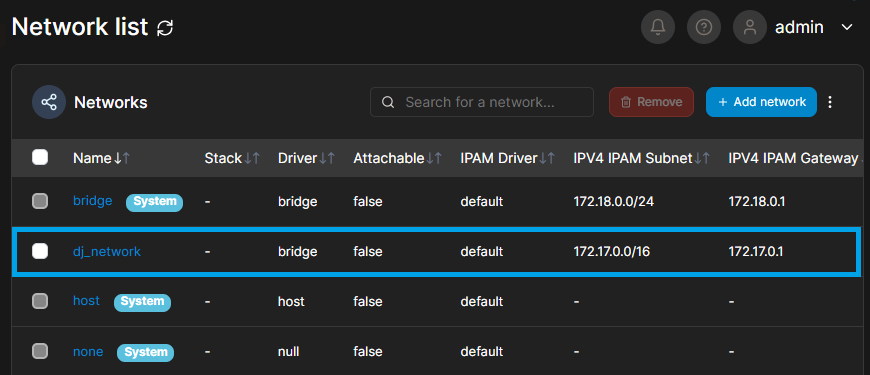
Результат выполнения команды docker ps -a отображен на рисунке 1:



***Рисунок 1 - Список контейнеров на основном сервере***

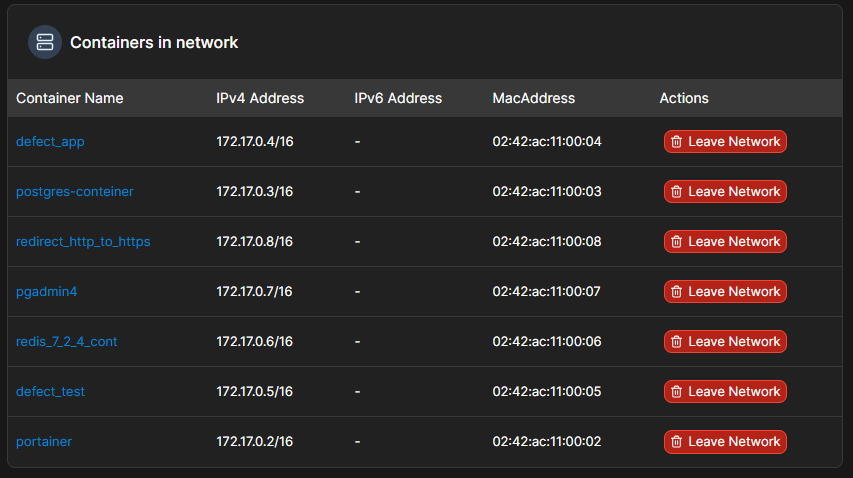
**1.3. Организация сети между контейнерами ВР «ЭЖД»**

Для организации взаимодействия между автономными контейнерами создана сеть **dj\_network** с сетевым драйвером bridge, подсеть по умолчанию **172.17.0.0/16**. Параметры сети отображены на рисунке 2:



***Рисунок 2 - Список сетей Docker на основном сервере***

По умолчанию контейнеры в сети **dj\_network** имеют статические IP-адреса, которые отображены на рисунке 3:



***Рисунок 3 - Список контейнеров в сети dj\_network***

**2. Обновление компонентов ВР «ЭЖД»**

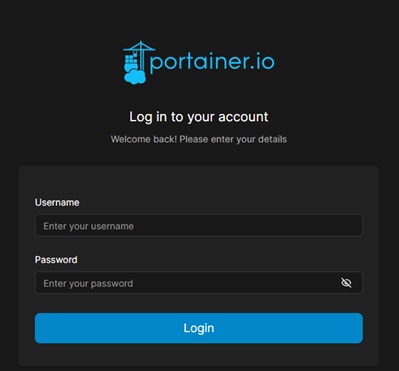
**2.1. Обновление приложения с использованием GUI**

Для обновления контейнера приложения defect\_app используйте графический интерфейс Portainer.

1. Вход в Portainer:

- Откройте веб-браузер и перейдите по адресу https://<URL\_для\_входа\_в\_portainer>/.

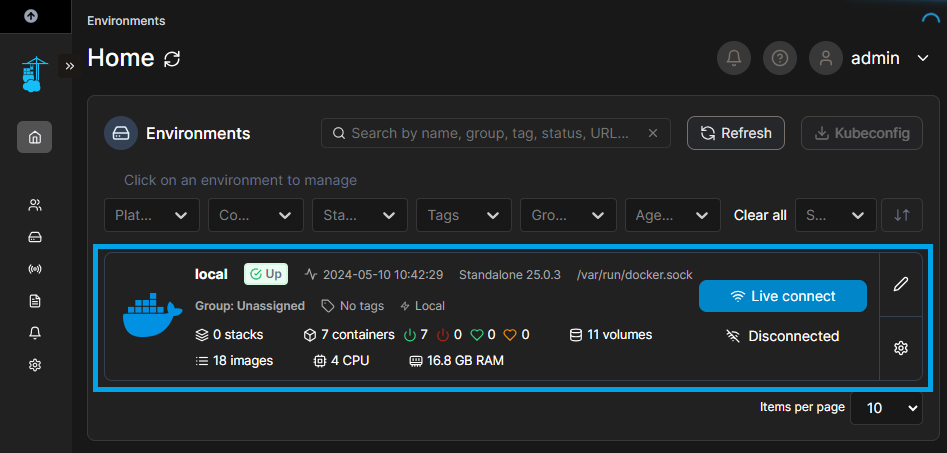
- Введите учетные данные для входа в систему (username и password).



***Рисунок 4 - Страница входа в приложение Portainer***

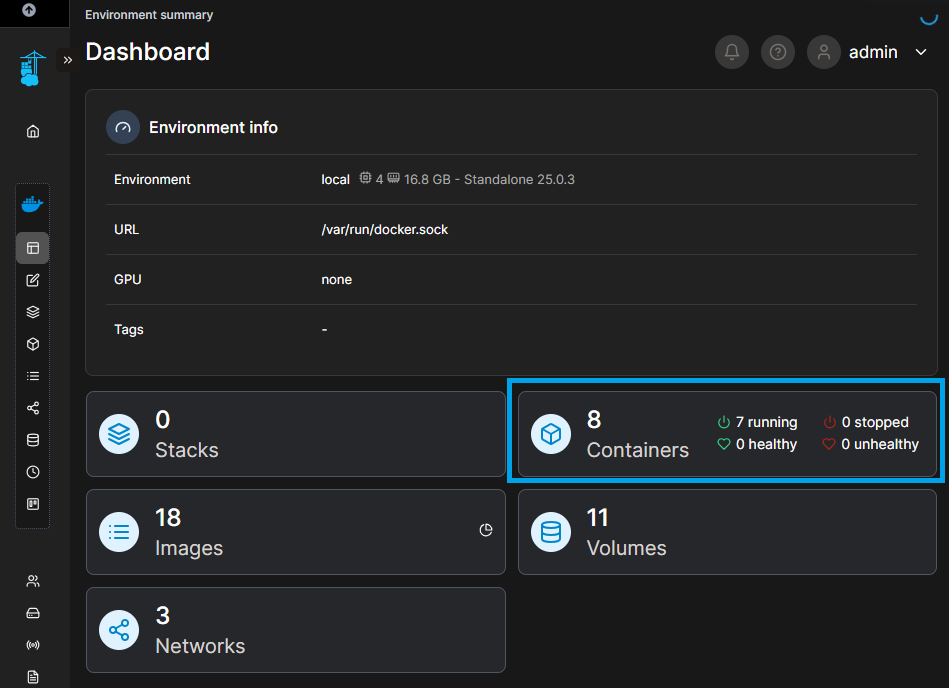
2. Выбор контейнера:

- Выберите «local» в списке Environments, как показано на рисунке 5.



***Рисунок 5 - Домашняя страница Portainer***

- Выберите «Containers» на Dashboard, как показано на рисунке 6.



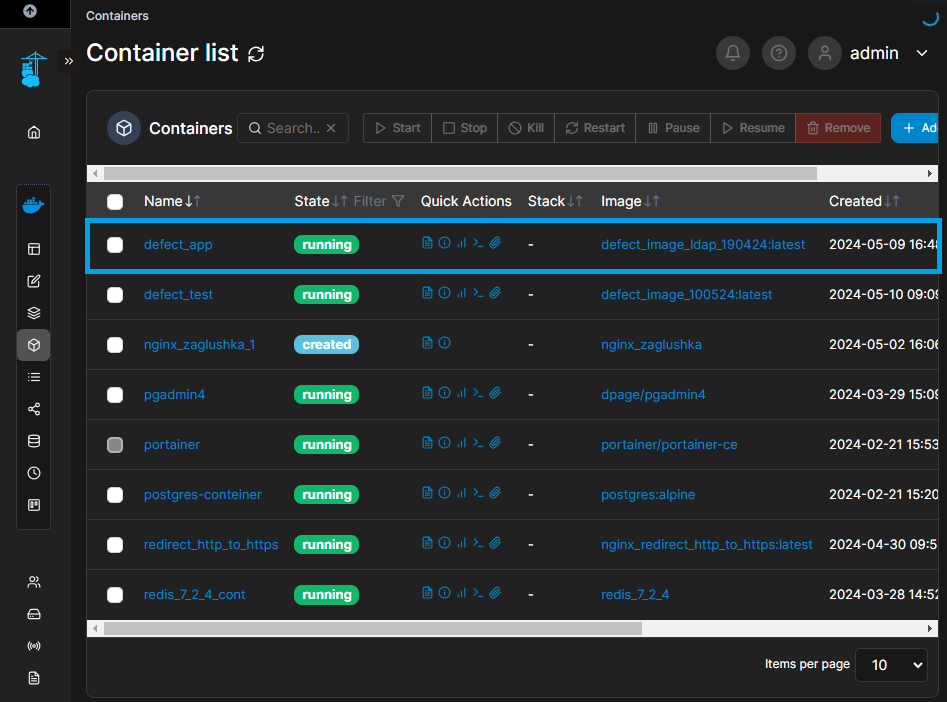
***Рисунок 6 - Информация об окружении***

3. Поиск контейнера:

На рисунке 7 представлены следующие кнопки для управления контейнерами:

* Кнопка «**Start**» - запускает выбранный контейнер, который в данный момент остановлен;
* Кнопка «**Stop**» - останавливает работающий контейнер, сохраняя его состояние для последующего запуска;
* Кнопка «**Kill**» - немедленно завершает работу контейнера без сохранения состояния, что может привести к потере данных;
* Кнопка «**Restart**» - перезапускает контейнер, сначала останавливая его, а затем снова запуская. Это полезно для обновления конфигурации или решения временных сбоев;
* Кнопка «**Pause**» - приостанавливает выполнение контейнера, сохраняет его в текущем состоянии. Контейнер остается в памяти, но не выполняет никакие процессы;
* Кнопка «**Resume**» - возобновляет выполнение приостановленного контейнера, позволяя ему продолжить работу с того же места;
* Кнопка «**Remove**» - удаляет контейнер из системы, включая все его данные и состояние. Необратимая операция;
* Кнопка «**Add Container**» - открывает форму для создания и запуска нового контейнера с необходимыми настройками и параметрами.

- Найдите соответствующий контейнер в Container list, как показано на рисунке 7.

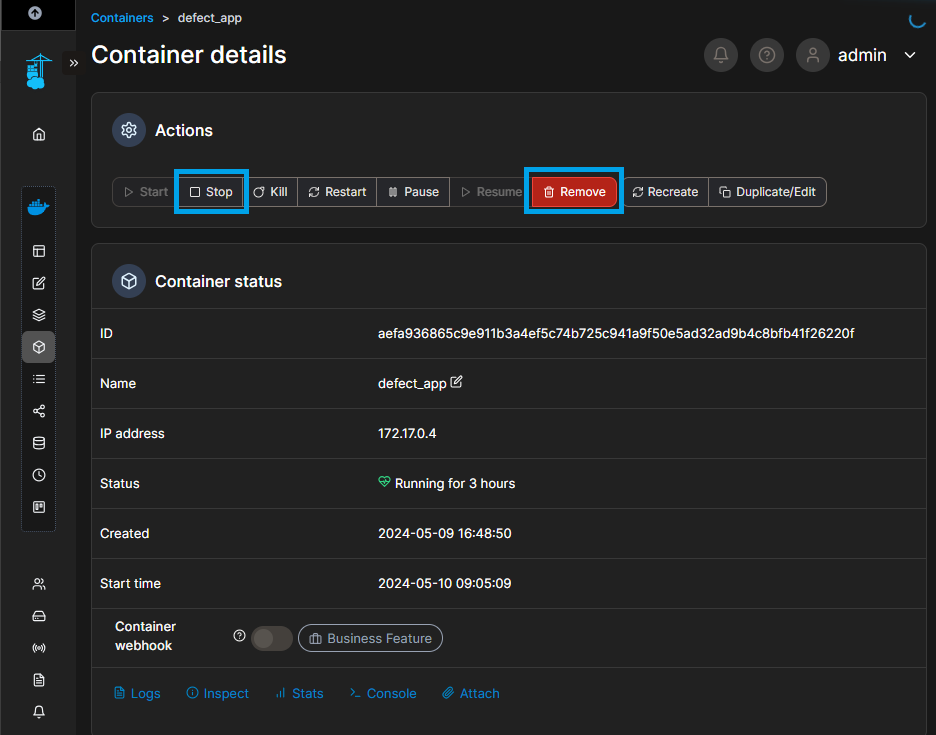


***Рисунок 7 - Список контейнеров окружения local***

4. Остановка и удаление контейнера:

- Нажмите на название контейнера, чтобы открыть его.

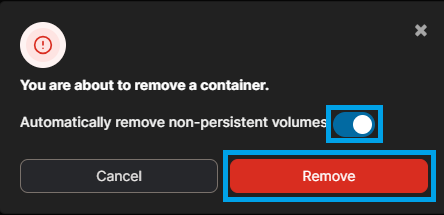
- Нажмите на кнопку «Stop», далее нажмите кнопку «Remove», чтобы остановить и удалить контейнер, как показано на рисунке 8.



***Рисунок 8 - Информация о контейнере***

5. Удаление контейнера:

- Если вы выбрали «Remove» в предыдущем шаге, подтвердите удаление контейнера, как показано на рисунке 9. Переключатель «Automatically remove non-persistent volumes» **активировать только если** том с данными от удаляемого контейнера больше не понадобится.



***Рисунок 9 - Диалоговое окно при удалении контейнера***

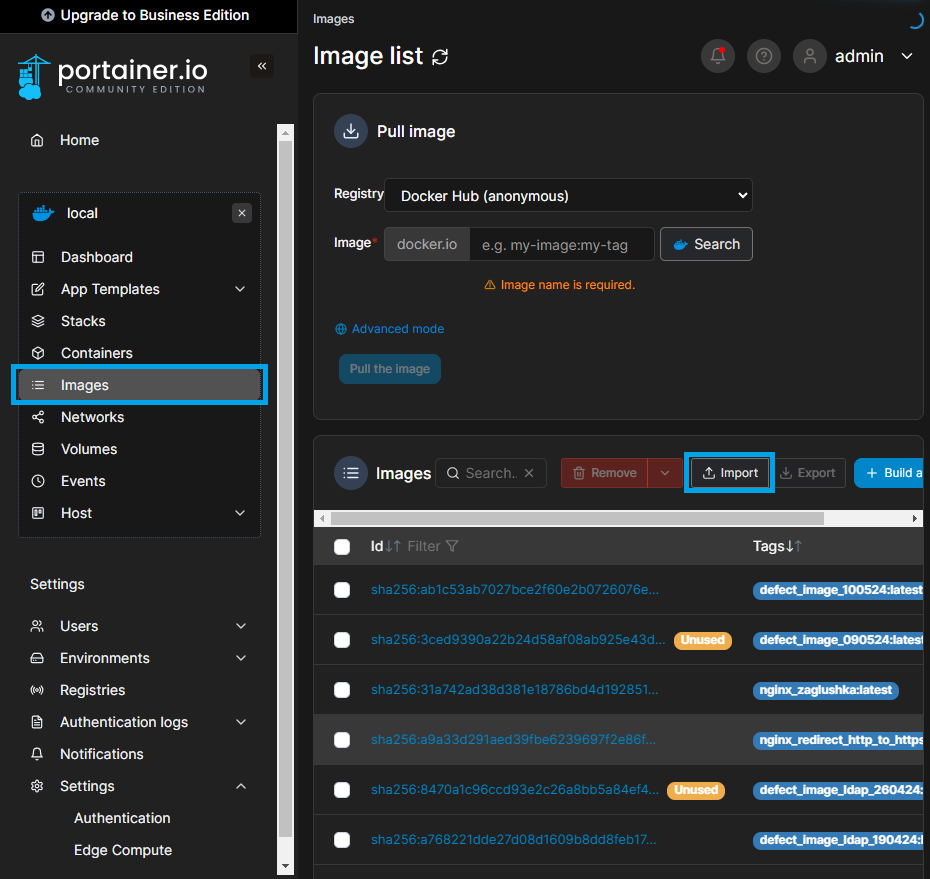
6. Загрузка нового образа:

На рисунке 10 представлены следующие кнопки для управления контейнерами:

* Кнопка «**Search**» - осуществляет поиск образов в Docker Hub или другом настроенном реестре. Позволяет находить и просматривать доступные образы для загрузки и использования;
* Кнопка «**Pull the image**» - загружает указанный образ из реестра Docker Hub или другого реестра образов в локальное хранилище. Этот процесс позволяет подготовить образ для последующего создания контейнеров;
* Кнопка «**Remove**» - удаляет выбранный образ из локального хранилища Docker. Полезно для освобождения места и управления ненужными или устаревшими образами;
* Кнопка «**Import**» - импортирует образ из локального файла архива (например, .tar). Этот процесс позволяет добавить образы, ранее сохраненные или экспортированные;
* Кнопка «**Export**» - экспортирует выбранный образ в файл архива (например, .tar). Полезно для сохранения образов для последующего импорта или резервного копирования;
* Кнопка «**Build a new image**» - создает новый образ из Dockerfile, который может быть загружен с локального устройства или из удаленного репозитория. Этот процесс собирает образ по указанным инструкциям в Dockerfile.

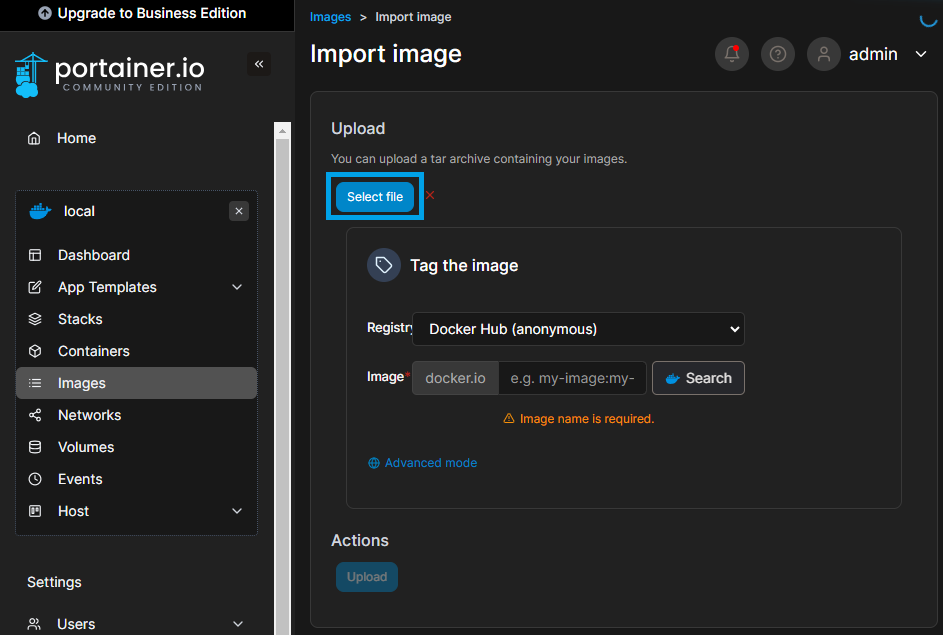
- В левом меню выберите «Images» как показано на рисунке 10.

- Нажмите на кнопку «Import» как показано на рисунке 10.



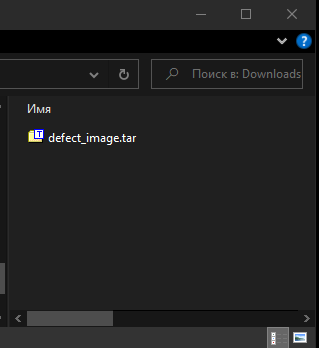
***Рисунок 10 - Список образов***

- Нажмите на кнопку «Select file» как показано на рисунке 11.



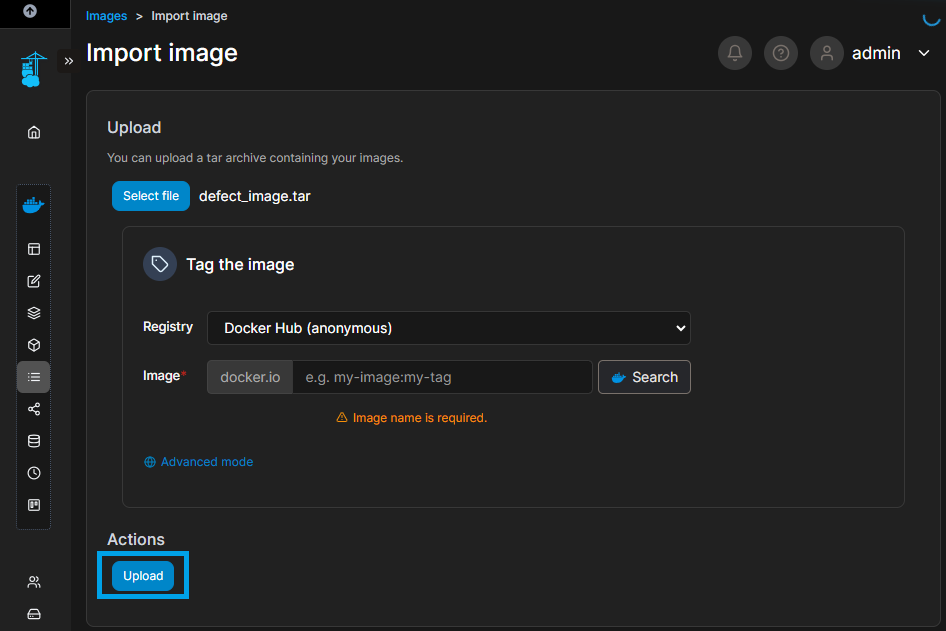
***Рисунок 11 - Импорт образа***

- Выберите файл defect\_image.tar (рисунок 12. Имя файла может отличаться), который содержит новую версию приложения.



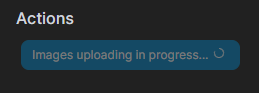
***Рисунок 12 - Архив с образом на рабочей станции с ОС Windows***

- Нажмите на кнопку «Upload» как показано на рисунке 13.



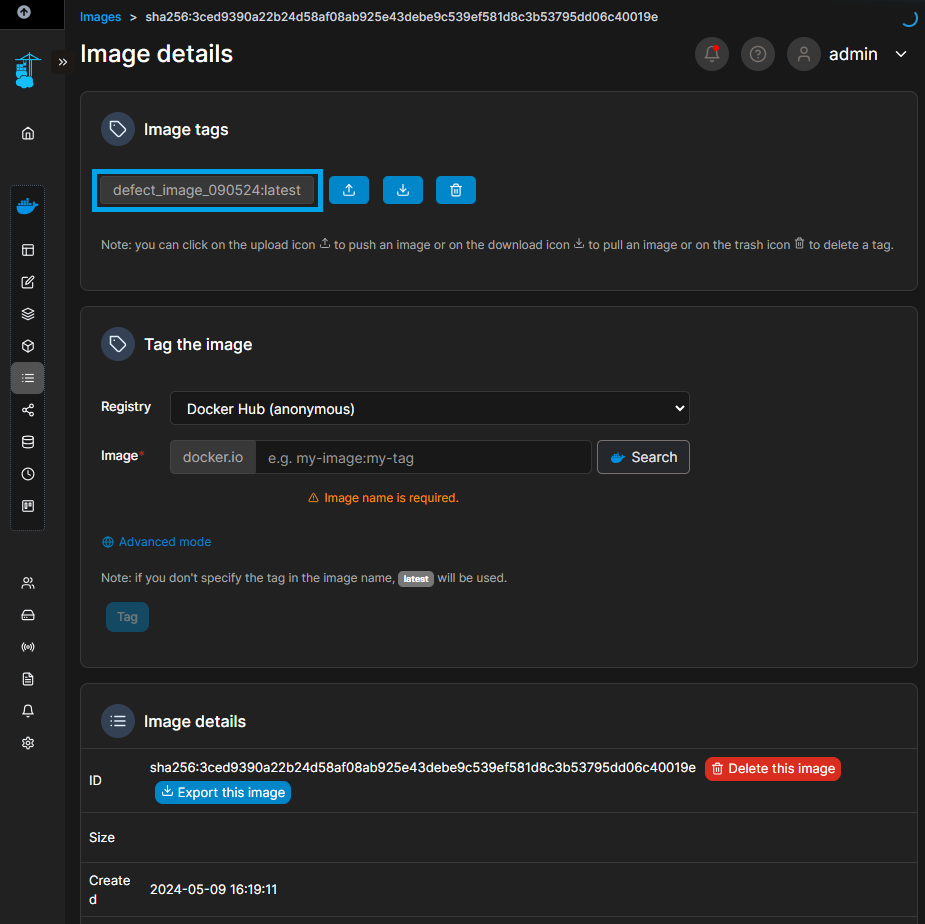
***Рисунок 13 - Загрузка выбранного образа***

- Дождитесь завершения загрузки, которая отображена на рисунке 14.



***Рисунок 14 - Процесс загрузки образа***

- Скопируйте Image Tag загруженного образа (он понадобится при создании контейнера). Пример с Image Tag отображен на рисунке 15. Также Image Tag нужного образа можно найти, нажав на Id из списка Images (см. пункт 6).

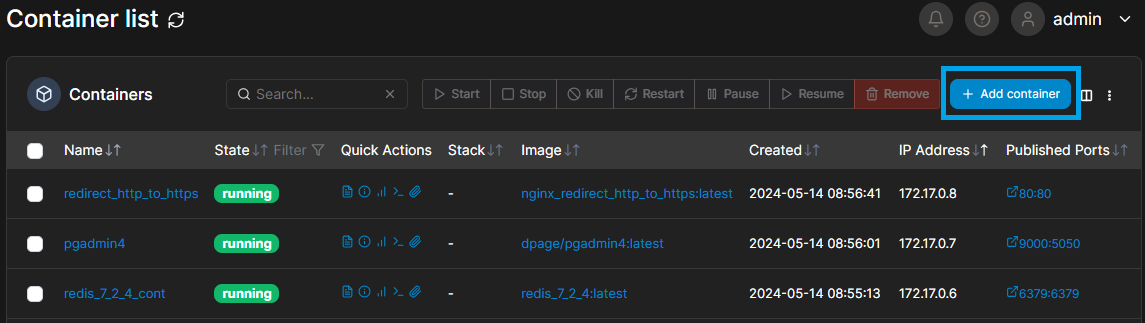


***Рисунок 15 - Информация о загруженном образе***

7. Создание нового контейнера:

- Вернитесь в раздел «Containers», который отображен на рисунке 16.

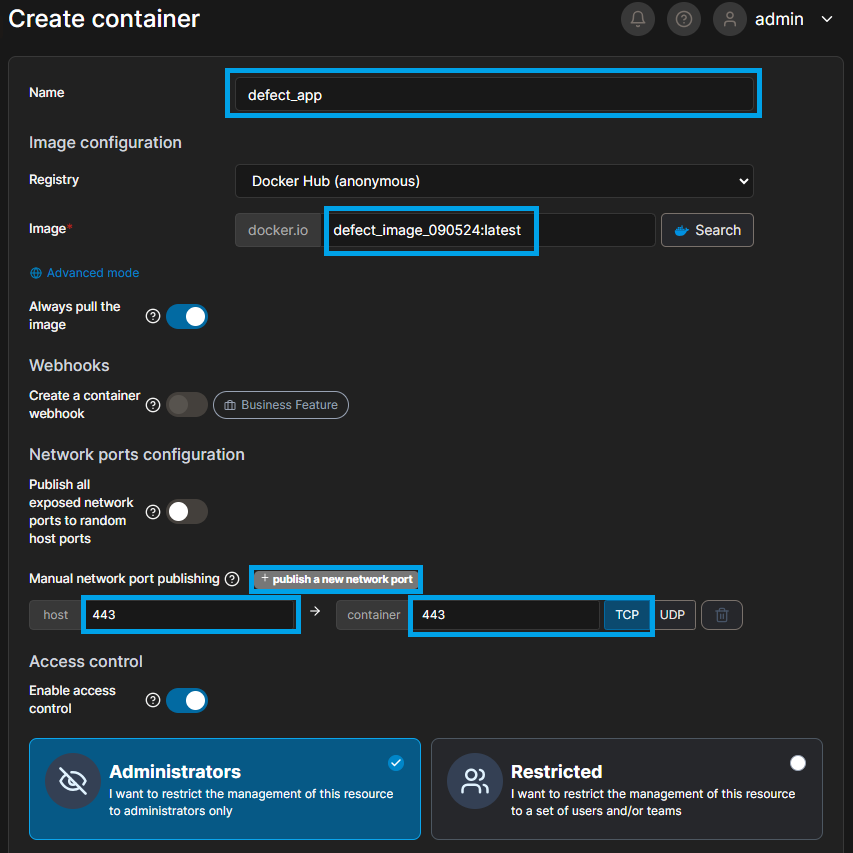
- Нажмите на кнопку «Add Container» как показано на рисунке 16.



***Рисунок 16 - Добавление контейнера***

- Укажите основные параметры (рисунок 17), где:

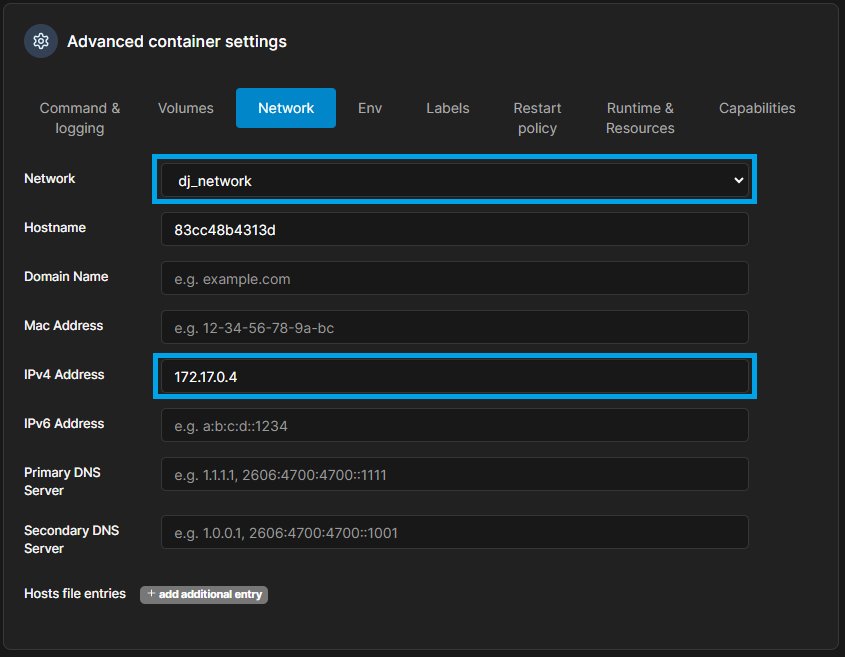
* «Name» - имя контейнера;
* «Image» - тэг образа (см. пункт 6);
* «Manual Network Port Publishing» - порты на хосте и в контейнере.



***Рисунок 17 - Основные параметры при создании контейнера***

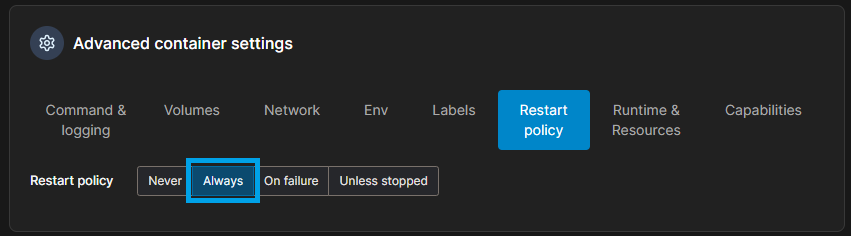
- Укажите дополнительные параметры (рисунок 18), которые находятся внизу страницы, где:

* Вкладка «Network» - настройки сети;
* «Network» - имя сети, выбрать из списка **dj**\_**network**;
* «IPv4 Address» - статический IP-адрес для контейнера приложения, задать IP-адрес по умолчанию.



***Рисунок 18 - Дополнительные параметры при создании контейнера***

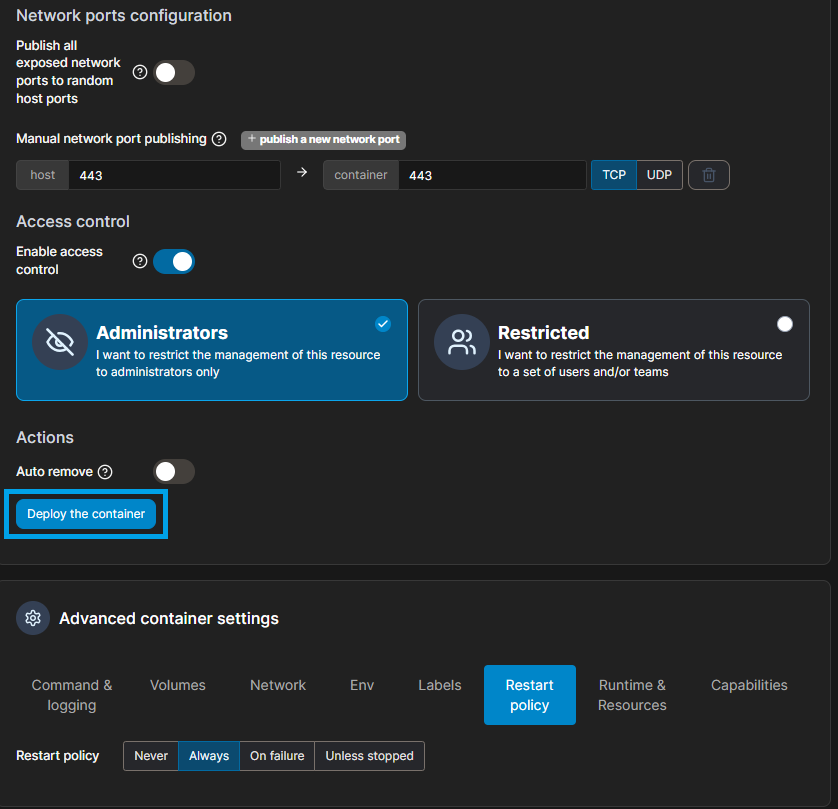
* Вкладка «Restart policy» (рисунок 19) - параметры перезапуска контейнера;
* «Restart policy» - выбрать Always, чтобы при перезапуске сервера / службы Docker контейнер с приложением запускался автоматически.



***Рисунок 19 - Параметры перезапуска контейнера***

8. Запуск нового контейнера:

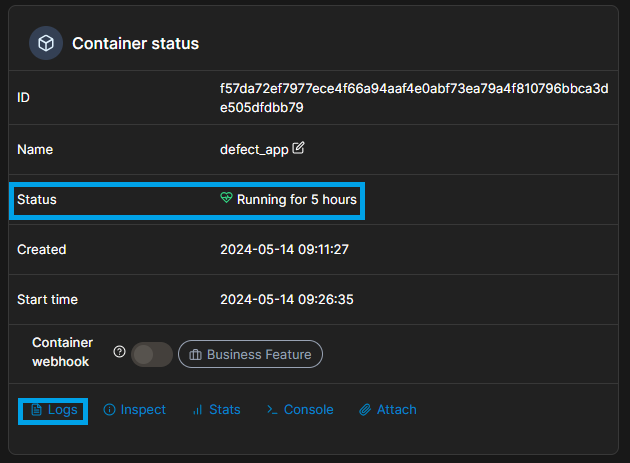
- Нажмите на кнопку «Deploy The Container», чтобы запустить новый контейнер c обновленной версией приложения как показано на рисунке 20.



***Рисунок 20 - Запуск нового контейнера***

9. Мониторинг процесса:

- После запуска нового контейнера следите за процессом его загрузки и запуска как показано на рисунках 20 и 21.



***Рисунок 20 - Информация о статусе контейнера***

C:\Users\t.petrov\Desktop\Безымянный.png

***Рисунок 21 - Текущий статус контейнера***

10. Проверка работоспособности:

- После успешного запуска контейнера убедитесь, что приложение работает корректно. Перейдите в браузере по адресу: https://<URL\_для\_входа\_в\_приложение>/ и авторизуйтесь под своей учетной записью.

**2.2. Обновление приложения с использованием CLI**

Для обновления образа контейнера приложения defect\_app через командную строку выполните следующие шаги:

1. Копирование образа приложения на основной сервер с рабочей станции с ОС Windows:

- Откройте командную строку на рабочей станции с ОС Windows.

- Используйте команду scp для копирования файла defect\_image.tar (имя архива с образом может отличаться) на основной сервер. Замените /path/to/defect\_image.tar на фактический путь к файлу defect\_image.tar на вашей рабочей станции, а username и /path/to/docker\_images/ на ваши учетные данные для доступа к основному серверу по SSH и путь к каталогу на основном сервере, куда будет скопирован образ:

**scp /path/to/defect**\_**image.tar username@ip\_адрес\_сервера:/path/to/docker**\_**images/**

где,

* **/path/to/defect\_image.tar** -путь к файлу, который вы хотите скопировать;
* **username** - имя пользователя с правами администратора на основном сервере;
* **ip\_адрес\_сервера** - IP-адрес основного сервера, на котором развернуто приложение;
* **:/path/to/docker\_images/** - путь на удаленном сервере, куда вы хотите скопировать файл.

- Введите пароль от учетной записи username для подтверждения копирования файла на сервер.

2. SSH-подключение к серверу:

- Откройте командную строку на вашей рабочей станции или воспользуйтесь программой для подключения к удаленному серверу (например, PuTTY).

- Используйте команду SSH для подключения к основному серверу:

**ssh username@ip\_адрес\_сервера**

где,

* **username** - имя пользователя с правами администратора на основном сервере;
* **ip\_адрес\_сервера** - IP-адрес основного сервера, на котором развернуто приложение.

- Введите пароль от учетной записи username для подключения к серверу.

3. Остановка контейнера:

- Используйте команду docker ps для просмотра запущенных контейнеров, имен образов и их ID.

- Выберите ID контейнера и имя образа, соответствующего имени defect\_app.

- Остановите контейнер с помощью команды:

**docker stop <container\_id>**

4. Удаление старого контейнера:

- Удалите контейнер с помощью команды:

**docker rm <container\_id>**

5. Удаление старого образа:

- Удалите образ с помощью команды:

**docker rmi <image\_id>**

6. Загрузка нового образа:

- Используйте команду **docker load** для загрузки нового образа:

**docker load -i /path/to/docker\_images/defect\_image.tar**

где,

* **-i** - флаг указывает Docker, что ввод для загрузки образа будет передан через стандартный поток ввода;
* **/path/to/docker\_images/defect\_image.tar -** путь к файлу, содержащему сохраненный образ Docker, который вы хотите загрузить.

7. Запуск нового контейнера:

- Используйте команду **docker run** для создания и запуска нового контейнера:

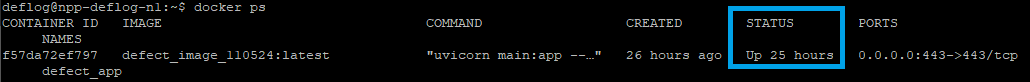
**docker run –d --name defect\_app –-network dj\_network -–ip <static\_container\_ip> –p 443:443 –-restart always <image\_name>**

где,

* **-d -** флаг, который указывает Docker на запуск контейнера в фоновом режиме;
* **--name -** опция для назначения контейнеру имени defect\_app;
* **--network** - опция для присоединения контейнера к сети dj\_network;
* **--ip <static\_container\_ip> -** опция для назначения контейнеру статического IP-адреса (см. рисунок 3) в сети dj\_network;
* **-p 443:443 -** опция для проброса портов. Пробрасывает порт 443 хоста на порт 443 контейнера, обеспечивая доступ к сервису через этот порт;
* **--restart always -** опция для автоматического перезапуска контейнера в случае его остановки;
* **<image\_name> -** имя образа Docker, на основе которого будет создан и запущен контейнер.

8. Проверка работоспособности:

- После успешного запуска нового контейнера, убедитесь, что приложение корректно работает, выполнив команду docker ps, результат выполнения которой отображен на рисунке 22, где в строке с контейнером defect\_app в колонке Status будет отображен статус контейнера.



***Рисунок 22 - Текущий статус контейнера в командной строке***

**3. Организация резервного копирования**

**3.1. Автоматическое резервное копирование БД**

По умолчанию на резервном сервере предустановлен скрипт postgres\_backup.sh, который по умолчанию настроен на ежедневное резервное копирование БД defectdb в 22:00 и хранение последних 20 резервных копий БД.

В случае необходимости изменения настроек, следует обратить внимание на следующие параметры:

* Путь до скрипта: /usr/local/bin/postgres\_backup.sh
* Путь до резервных копий БД: /var/backups/postgres

Содержимое скрипта postgres\_backup.sh:

#!/bin/bash

current\_date=$(date +%d%m%Y)

ssh username@ip\_адрес\_сервера "docker exec postgres-conteiner bash -c \"pg\_dump -U postgres -F t defectdb\"" > /var/backups/postgres/backup\_${current\_date}.tar

find /var/backups/postgres/backup\_\* -mtime +20 -exec rm {} \;

где,

* **current\_date** - переменная используется для создания даты в формате %d%m%Y, которая включается в имя файла резервной копии;
* **username** - имя пользователя с правами администратора на основном сервере;
* **ip\_адрес\_сервера** - IP-адрес основного сервера, на котором развернута БД;
* **docker exec postgres-conteiner bash -c "pg\_dump -U postgres -F t defectdb" -** команда выполняет дамп БД defectdb в формате .tar на основном сервере с использованием Docker контейнера с именем postgres-conteiner;
* **> /var/backups/postgres/backup\_${current\_date}.tar -** результат выполнения команды pg\_dump записывается в файл резервной копии с именем в формате backup\_${current\_date}.tar в каталоге /var/backups/postgres/;
* **find /var/backups/postgres/backup\_\* -** команда используется для поиска файлов в каталоге /var/backups/postgres/, имена которых начинаются с backup\_;
* **-mtime +20 -** параметр указывает find на поиск файлов, которые были изменены более 20 дней назад. В данном случае, это означает что find будет искать файлы, которые старше 20 дней;
* **-exec rm {} \; -** параметр find используется для выполнения команды rm (удаление) для каждого найденного файла. {} представляет собой заменитель для имени найденного файла, а \; указывает на конец команды exec. Таким образом, для каждого найденного файла, который старше 20 дней, будет выполнена команда rm, чтобы удалить его.

Для просмотра расписания в системе cron на резервном сервере выполните в командной строке crontab -e. Это откроет редактор crontab для текущего пользователя, в котором настроено следующее расписание:

0 22 \* \* \* /usr/local/bin/postgres\_backup.sh

где,

* **0 22 \* \* \*** - расписание cron, которое указывает на то, что скрипт должен запускаться каждый день в 22:00;
* **/usr/local/bin/postgres\_backup.sh** - путь к скрипту, который должен выполняться по расписанию.

Убедитесь, что вы выполняете эти действия от имени пользователя, для которого вы хотите настроить crontab.

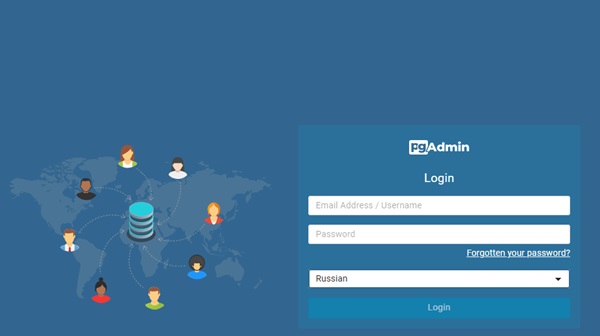
**3.2. Ручное резервное копирование БД**

**3.2.1. Резервное копирование БД с использованием pgAdmin**

1. Вход в pgAdmin отображен на рисунке 23:

- Откройте веб-браузер и перейдите по адресу: https://<URL\_для\_входа\_в\_pgAdmin>/.

- Введите учетные данные администратора для входа в систему.



***Рисунок 23 - Страница входа в приложение pgAdmin***

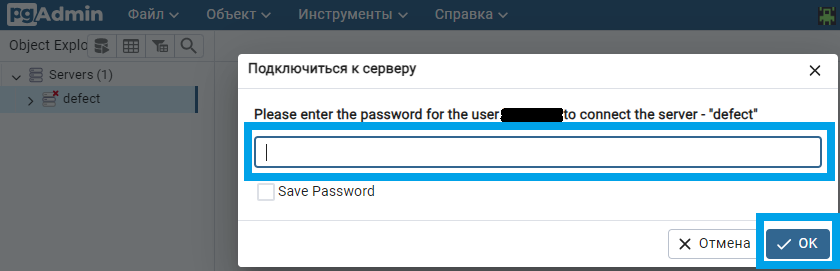
2. Подключение к серверу БД отображено на рисунке 24:

- В левой панели навигации выберите «Servers».

- Выберите defect.

- Введите пароль для подключения к серверу БД.

- Нажмите «OK».

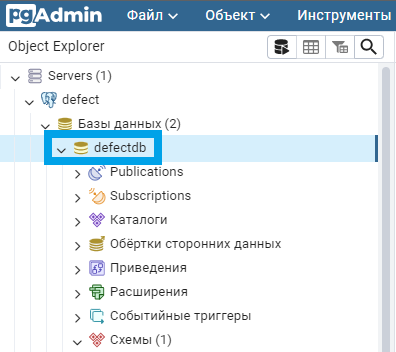


***Рисунок 24 - Подключение к серверу БД***

3. Выбор БД:

- Разверните дерево сервера, чтобы увидеть доступные БД как показано на рисунке 25.

- Найдите БД defectdb в списке БД.

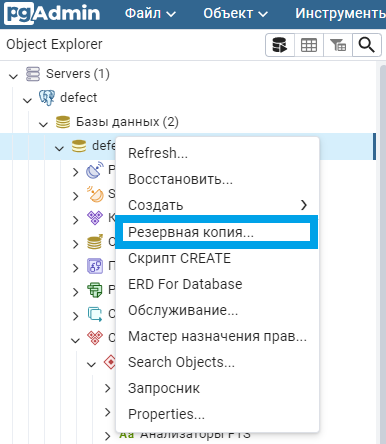


***Рисунок 25 - Список с БД***

4. Создание резервной копии:

- Щелкните правой кнопкой мыши на базе данных defectdb.

- В контекстном меню выберите «Резервная копия» как показано на рисунке 26.



***Рисунок 26 - Контекстное меню БД***

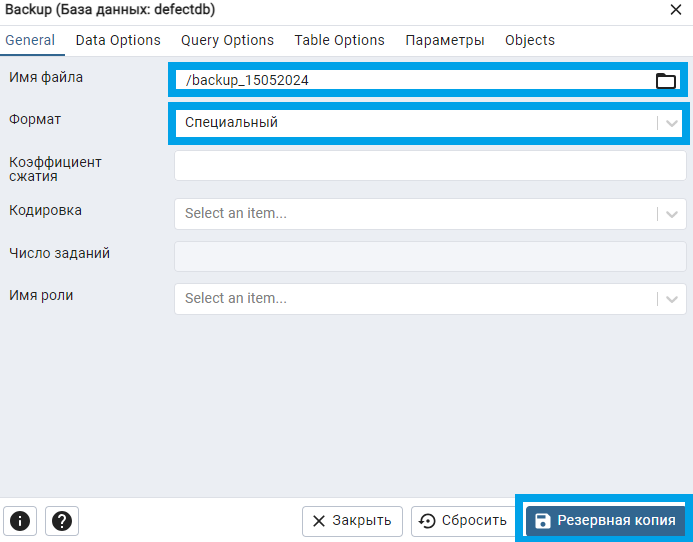
5. Настройка параметров резервной копии, которые показаны на рисунке 27:

- Укажите имя файла, под которым будет сохранена резервная копия.

- Выберите формат резервной копии «Специальный» или «Tar».

6. Запуск процесса создания резервной копии:

- Нажмите кнопку Резервная копия, как показано на рисунке 27, чтобы запустить процесс создания резервной копии.

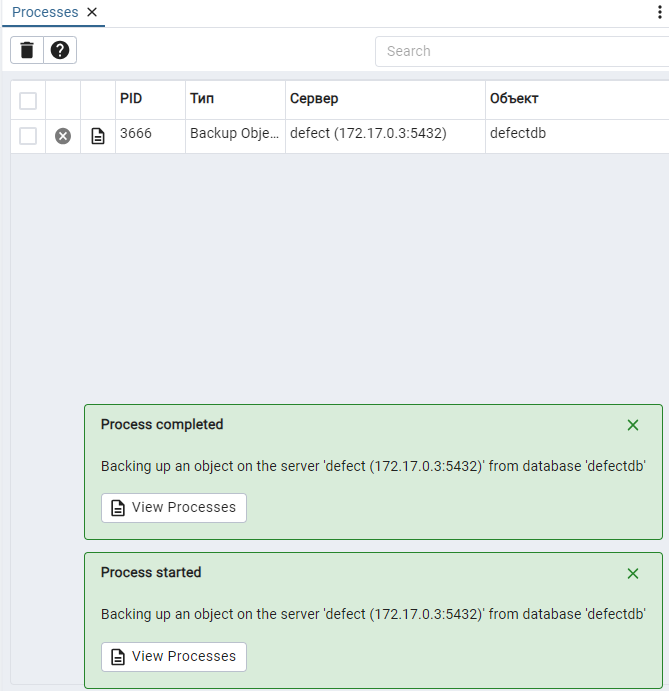


***Рисунок 27 - Параметры резервной копии БД***

7. Проверка успешного выполнения как отображено на рисунке 28:

- Дождитесь завершения процесса создания резервной копии.

- Проверьте, что файл резервной копии был создан в указанном месте.



***Рисунок 28 - Статусы создания резервной копии БД***

**3.2.2. Резервное копирование БД с использованием pg\_dump**

1. SSH-подключение к серверу:

- Откройте командную строку на вашей рабочей станции или воспользуйтесь программой для подключения к удаленному серверу (например, PuTTY).

**ssh username@ip\_адрес\_сервера**

где,

* **username** - имя пользователя с правами администратора на основном сервере;
* **ip\_адрес\_сервера** - IP-адрес основного сервера, на котором развернута БД.

- Введите пароль от учетной записи username для подключения к серверу.

2. Вход в контейнер PostgreSQL:

- Используйте команду docker exec для входа в контейнер PostgreSQL.

**docker exec -it postgres-conteiner bash**

где,

* **-i** - флаг указывает Docker на то, что нужно держать стандартный ввод открытым, что позволяет взаимодействовать с процессом внутри контейнера;
* **-t** - флаг указывает Docker на то, что нужно выделить псевдотерминал. В сочетании с флагом **-i** позволяет запускать команду в интерактивном режиме, чтобы можно было работать с оболочкой так же, как в обычном режиме;
* **postgres-conteiner** - имя контейнера, в котором будет выполняться команда;
* **bash** - команда, которую нужно выполнить внутри контейнера. В данном случае открывается Bash-оболочка, позволяющая пользователю интерактивно работать внутри контейнера.

3. Создание резервной копии БД:

- Внутри контейнера выполните команду pg\_dump.

**pg\_dump -U postgres -F t defectdb > /var/backups/postgres/backup.tar**

где,

* **-U postgres -** флаг -U указывает имя пользователя БД, которое будет использоваться для подключения к PostgreSQL. В данном случае, используется пользователь postgres;
* **-F t** - флаг -F определяет формат резервной копии. t означает tar архив, который создает резервную копию в формате tar-архива;
* **defectdb** - имя БД, для которой создается резервная копия;
* **> /var/backups/postgres/backup.tar** - оператор перенаправления вывода > указывает, что результат команды (резервная копия базы данных) должен быть записан в файл /var/backups/postgres/backup.tar.

4. Выход из контейнера PostgreSQL:

- Если вы закончили работу в контейнере, нажмите комбинацию клавиш «Ctrl + D», чтобы выйти из контейнера.

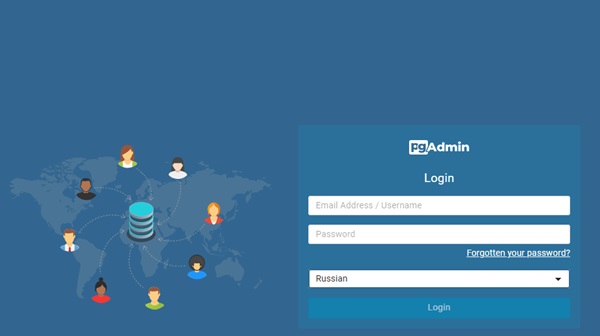
**4****. Восстановление БД из резервной копии**

**4****.1. Восстановление БД с использованием pgAdmin**

1. Вход в pgAdmin отображен на рисунке 29:

- Откройте веб-браузер и перейдите по адресу: https://<URL\_для\_входа\_в\_pgAdmin>/.

- Введите учетные данные администратора для входа в систему.



***Рисунок 29 - Страница входа в приложение pgAdmin***

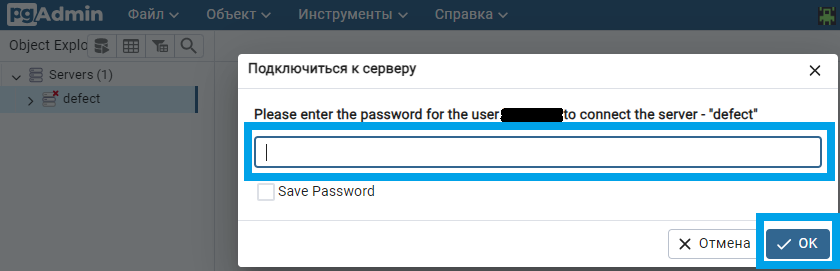
2. Подключение к серверу БД отображено на рисунке 30:

- В левой панели навигации выберите «Servers».

- Выберите defect.

- Введите пароль для подключения к серверу БД.

- Нажмите «OK».

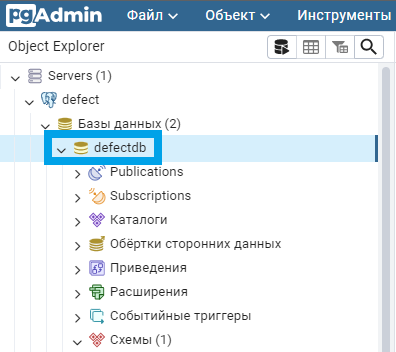


***Рисунок 30 - Подключение к серверу БД***

3. Выбор БД:

- Разверните дерево сервера, чтобы увидеть доступные БД как показано на рисунке 31.

- Найдите БД defectdb в списке БД.

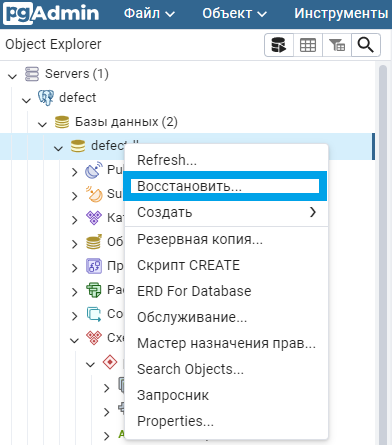


***Рисунок 31 - Список с БД***

4. Выбор опции восстановления:

- Щелкните правой кнопкой мыши на базе данных defectdb.

- В контекстном меню выберите «Восстановить» как показано на рисунке 32.



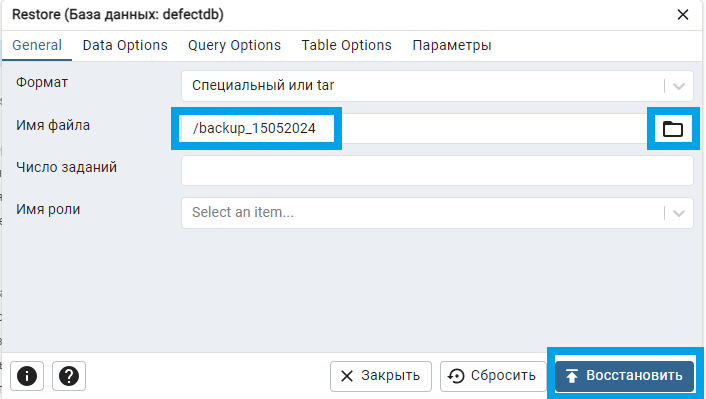
***Рисунок 32 - Контекстное меню БД***

5. Выбор файла резервной копии:

- В окне восстановления базы данных, укажите путь к файлу резервной копии как показано на рисунке 33.

6. Запуск процесса восстановления:

- Нажмите кнопку «Восстановить» как показано на рисунке 33, чтобы запустить процесс восстановления базы данных из резервной копии.



***Рисунок 33 - Параметры восстановления резервной копии БД***

**4****.2. Восстановление БД с использованием pg\_restore**

1. SSH-подключение к серверу:

- Откройте командную строку на вашей рабочей станции или воспользуйтесь программой для подключения к удаленному серверу (например, PuTTY).

**ssh username@ip\_адрес\_сервера**

где,

* **username** - имя пользователя с правами администратора на основном сервере;
* **ip\_адрес\_сервера** - IP-адрес основного сервера, на котором развернута БД.

- Введите пароль от учетной записи username для подключения к серверу.

2. Вход в контейнер PostgreSQL:

- Используйте команду docker exec для входа в контейнер PostgreSQL.

**docker exec -it postgres-conteiner bash**

где,

* **-i** - флаг указывает Docker на то, что нужно держать стандартный ввод открытым, что позволяет взаимодействовать с процессом внутри контейнера;
* **-t** - флаг указывает Docker на то, что нужно выделить псевдотерминал. В сочетании с флагом **-i** позволяет запускать команду в интерактивном режиме, чтобы можно было работать с оболочкой так же, как в обычном режиме;
* **postgres-conteiner** - имя контейнера, в котором будет выполняться команда;
* **bash** - команда, которую нужно выполнить внутри контейнера. В данном случае открывается Bash-оболочка, позволяющая пользователю интерактивно работать внутри контейнера.

3. Восстановление БД с помощью pg\_restore:

- В контейнере postgres-conteiner используйте команду pg\_restore для восстановления базы данных defectdb из резервной копии в формате .tar.

**pg\_restore -U postgres -d defectdb /path/to/backup\_file.tar**

где,

* **-U postgres -** флаг -U указывает имя пользователя БД, которое будет использоваться для подключения к PostgreSQL. В данном случае, используется пользователь postgres;
* **-d defectdb** - флаг -d указывает имя целевой БД, в которую будет производиться восстановление. В данном случае, это БД defectdb;
* **/path/to/backup\_file.tar** - путь к файлу резервной копии, созданной ранее с помощью pg\_dump и содержащей данные, которые нужно восстановить. В данном случае, это файл backup\_file.tar

Путь к файлу резервной копии /path/to/backup\_file.tar замените на фактический путь к вашему файлу резервной копии.

4. Выход из контейнера PostgreSQL:

- Если вы закончили работу в контейнере, нажмите комбинацию клавиш «Ctrl + D», чтобы выйти из контейнера.

**5. Хранение резервных копий**

**5.1. Хранение резервных копий БД, созданных автоматически**

Ежедневные резервные копии БД хранятся на резервном сервере по пути /var/backups/postgres. По умолчанию, резервные копии хранятся за последние 20 дней. Этот параметр можно изменить в скрипте postgres\_backup.sh, в следующей строке (подробную информацию о скрипте см. в пункте 3.1 данного руководства):

**find /var/backups/postgres/backup\_\* -mtime +20 -exec rm {} \;**

Значение **+20** указывает количество дней хранения резервных копий. Для изменения периода хранения, замените это число на нужное вам количество дней.

**5.2. Хранение резервных копий БД, созданных вручную**

Место хранения резервных копий БД созданных вручную определяет системный администратор на свое усмотрение.

**5.3. Хранение резервных копий контейнеров и образов ВР «ЭЖД», созданных вручную**

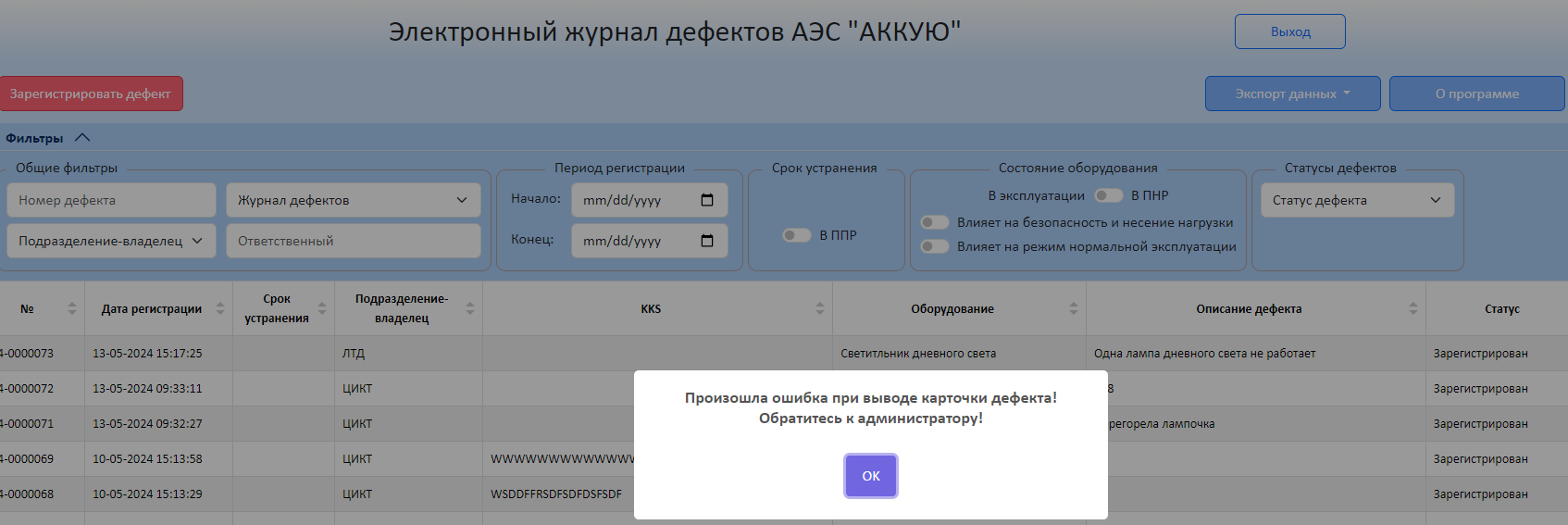
Место хранения резервных копий контейнеров и образов ВР «ЭЖД» определяет системный администратор на свое усмотрение.

**6. Устранение ошибок при работе с ВР «ЭЖД»**

**6.1. Ошибка при авторизации пользователя в ВР «ЭЖД»**

1. Пользователь не добавлен в подразделение журнала дефектов в Active Directory:
   * Пользователь должен убедиться, что ему согласован доступ к ВР «ЭЖД».
   * В случае согласованного доступа к ВР ЭЖД, пользователь должен обратиться в УИТ по внутреннему номеру телефона 777 для добавления его доменной учетной записи в необходимые группы в Active Directory.
2. Пользователь не может авторизоваться после добавления в необходимые группы в Active Directory:

При попытке авторизоваться пользователь видит окно с ошибкой, которое показано на рисунке 34:



***Рисунок 34 - Окно с ошибкой при авторизации пользователя в ВР «ЭЖД»***

Решение проблемы, которая отображена на рисунке 34:

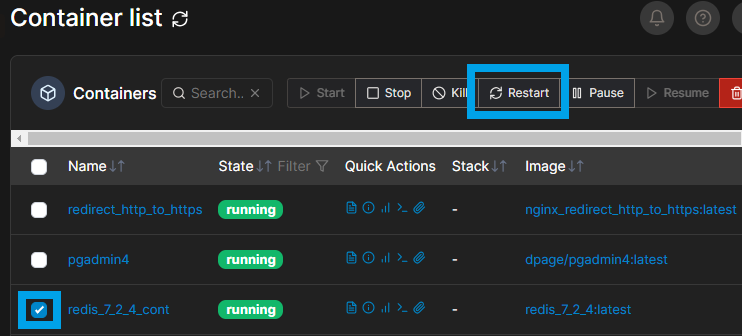
- Перезапустите контейнер redis\_7\_2\_4\_cont для обновления данных в кэше.

* Перезапуск с помощью Portainer (см. пункт 2.1 данного руководства):

- Авторизуйтесь в Portainer (см. рисунок 4).

- Перейдите в список контейнеров (см. рисунок 7).

- Выберите в списке контейнер с именем redis\_7\_2\_4\_cont и нажмите кнопку «Restart» как показано на рисунке 35.



***Рисунок 35 - Перезапуск контейнера redis\_7\_2\_4\_cont***

* Перезапуск с помощью CLI на основном сервере:

1. SSH-подключение к серверу:

- Откройте командную строку на вашей рабочей станции или воспользуйтесь программой для подключения к удаленному серверу (например, PuTTY).

- Используйте команду SSH для подключения к основному серверу:

**ssh username@ip\_адрес\_сервера**

где,

* **username** - имя пользователя с правами администратора на основном сервере;
* **ip\_адрес\_сервера** - IP-адрес основного сервера, на котором развернуто приложение.

- Введите пароль от учетной записи username для подключения к серверу.

2. Перезапуск контейнера:

- Перезапустите контейнер с помощью команды:

**docker restart redis\_7\_2\_4\_cont**

где,

* **redis\_7\_2\_4\_cont** - имя контейнера, которому требуется перезапуск.

3. Мониторинг процесса:

- После перезапуска контейнера следите за процессом его запуска как показано на рисунках 20 и 21.

**6.2. Запуск новых контейнеров через CLI в случае ошибок на текущих контейнерах в ВР «ЭЖД»**

В случае проблем с существующими контейнерами вместо них можно создать новые с теми же параметрами. Проблемный контейнер перед пересозданием необходимо удалить с помощью команды docker rm <container\_name>.

Приведенные в пунктах 6.2.1, 6.2.2, 6.2.3, 6.2.4 команды выполняются на основном сервере.

SSH-подключение к серверу:

- Откройте командную строку на вашей рабочей станции или воспользуйтесь программой для подключения к удаленному серверу (например, PuTTY).

- Используйте команду SSH для подключения к основному серверу:

**ssh username@ip\_адрес\_сервера**

где,

* **username** - имя пользователя с правами администратора на основном сервере;
* **ip\_адрес\_сервера** - IP-адрес основного сервера, на котором развернуты контейнеры с компонентами приложения.

- Введите пароль от учетной записи username для подключения к серверу.

**6.2.1. Создание нового контейнера defect\_app**

**docker run –d --name defect\_app –-network dj\_network -–ip <static\_container\_ip> –p 443:443 –-restart always <image\_name>**

где,

* **-d -** флаг, который указывает Docker на запуск контейнера в фоновом режиме;
* **--name -** опция для назначения контейнеру имени defect\_app;
* **--network** - опция для присоединения контейнера к сети dj\_network;
* **--ip <static\_container\_ip> -** опция для назначения контейнеру статического IP-адреса (см. рисунок 3) в сети dj\_network;
* **-p 443:443 -** опция для проброса портов. Пробрасывает порт 443 хоста на порт 443 контейнера, обеспечивая доступ к сервису через этот порт;
* **--restart always -** опция для автоматического перезапуска контейнера в случае его остановки;
* **<image\_name> -** имя образа Docker, на основе которого будет создан и запущен контейнер.

**6.2.2. Создание нового контейнера postgres-conteiner**

**docker run -d --name postgres-conteiner --network dj\_network --ip <static\_container\_ip> -p 5432:5432 -e POSTGRES\_PASSWORD=<userpassword> --restart always <image\_name>**

где,

* **-d -** флаг, который указывает Docker на запуск контейнера в фоновом режиме;
* **--name -** опция для назначения контейнеру имени postgres-conteiner;
* **--network** - опция для присоединения контейнера к сети dj\_network;
* **--ip <static\_container\_ip> -** опция для назначения контейнеру статического IP-адреса (см. рисунок 3) в сети dj\_network;
* **-p 5432:5432 -** опция для проброса портов. Пробрасывает порт 5432 хоста на порт 5432 контейнера, обеспечивая доступ к сервису через этот порт;
* **-e POSTGRES\_PASSWORD=<userpassword> -** опция для задания переменной окружения POSTGRES\_PASSWORD. Эта переменная устанавливает пароль для пользователя postgres в БД PostgreSQL, запущенной в контейнере. Значение userpassword следует заменить фактическим паролем;
* **--restart always -** опция для автоматического перезапуска контейнера в случае его остановки;
* **<image\_name> -** имя образа Docker, на основе которого будет создан и запущен контейнер.

**6.2.3. Создание нового контейнера pgadmin4**

**docker run --name pgadmin4 –-network dj\_network -–ip <static\_container\_ip> -p 9000:5050 -e "PGADMIN\_DEFAULT\_EMAIL=<useremail>" -e "PGADMIN\_DEFAULT\_PASSWORD=<userpassword>" -e "PGADMIN\_LISTEN\_ADDRESS=0.0.0.0" -e "PGADMIN\_LISTEN\_PORT=5050" –-restart always -d <image\_name>**

где,

* **--name -** опция для назначения контейнеру имени pgadmin4;
* **--network** - опция для присоединения контейнера к сети dj\_network;
* **--ip <static\_container\_ip> -** опция для назначения контейнеру статического IP-адреса (см. рисунок 3) в сети dj\_network;
* **-p 9000:5050 -** опция для проброса портов. Пробрасывает порт 9000 хоста на порт 5050 контейнера, обеспечивая доступ к сервису через этот порт;
* **-e "PGADMIN\_DEFAULT\_EMAIL=<useremail>" -** опция для задания переменной окружения PGADMIN\_DEFAULT\_EMAIL. Эта переменная устанавливает email по умолчанию для аутентификации в pgAdmin. Значение <useremail>следует заменить фактическим email;
* **-e "PGADMIN\_DEFAULT\_PASSWORD=<userpassword>" -** опция для задания переменной окружения PGADMIN\_DEFAULT\_PASSWORD. Эта переменная устанавливает пароль по умолчанию для аутентификации в pgAdmin. Значение <userpassword> следует заменить фактическим паролем;
* **-e "PGADMIN\_LISTEN\_ADDRESS=0.0.0.0" -** опция для задания переменной окружения PGADMIN\_LISTEN\_ADDRESS. Эта переменная устанавливает адрес прослушивания pgAdmin. Значение 0.0.0.0 означает прослушивание на всех доступных сетевых интерфейсах;
* **-e "PGADMIN\_LISTEN\_PORT=5050" -** опция для задания переменной окружения PGADMIN\_LISTEN\_PORT. Эта переменная устанавливает порт прослушивания pgAdmin. В данном случае, устанавливается порт 5050;
* **--restart always -** опция для автоматического перезапуска контейнера в случае его остановки;
* **-d -** флаг, который указывает Docker на запуск контейнера в фоновом режиме;
* **<image\_name> -** имя образа Docker, на основе которого будет создан и запущен контейнер.

**6.2.4. Создание нового контейнера redis\_7\_2\_4\_cont**

**docker run -d --name redis\_7\_2\_4\_cont –-network dj\_network -–ip <static\_container\_ip> -p 6379:6379 –-restart always <image\_name>**

где,

* **-d -** флаг, который указывает Docker на запуск контейнера в фоновом режиме;
* **--name -** опция для назначения контейнеру имени redis\_7\_2\_4\_cont;
* **--network** - опция для присоединения контейнера к сети dj\_network;
* **--ip <static\_container\_ip> -** опция для назначения контейнеру статического IP-адреса (см. рисунок 3) в сети dj\_network;
* **-p 6379:6379 -** опция для проброса портов. Пробрасывает порт 6379 хоста на порт 6379 контейнера, обеспечивая доступ к сервису через этот порт;
* **--restart always -** опция для автоматического перезапуска контейнера в случае его остановки;
* **<image\_name> -** имя образа Docker, на основе которого будет создан и запущен контейнер.