|  |
| --- |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение высшего образования **«МИРЭА Российский технологический университет»**  **РТУ МИРЭА** |

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА**

по направлению дополнительной программы профессиональной переподготовки

**«Технологии Девопс»**

**Тема работы**: «Тема берется из списка тем»

|  |  |  |
| --- | --- | --- |
| Обучающийся: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (Подпись) | студент Д.О. Враженко |
| Руководитель работы | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (Подпись) | доцент А.Т. Тарланов |

Москва 2025 г.

**Оглавление**

[Модуль 6. Администрирование баз данных 3](#__RefHeading___Toc1265_3933951894)

[Модуль 8. Мониторинг, логирование и оповещение событий 7](#__RefHeading___Toc1267_3933951894)

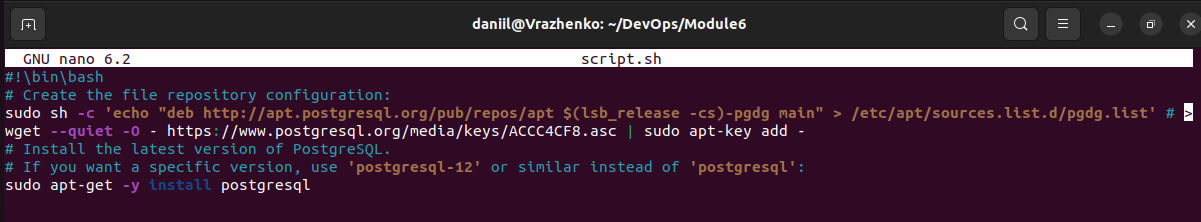
[Модуль 9. Виртуализация в DevOps 13](#__RefHeading___Toc1719_1609335944)

[Модуль 14. Жизненный цикл ПО 15](#__RefHeading___Toc1723_1609335944)

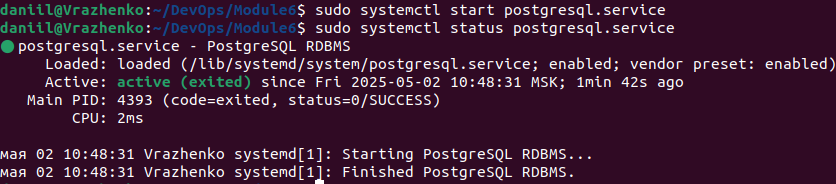
[Вывод 23](#__RefHeading___Toc1725_1609335944)

# Модуль 6. Администрирование баз данных

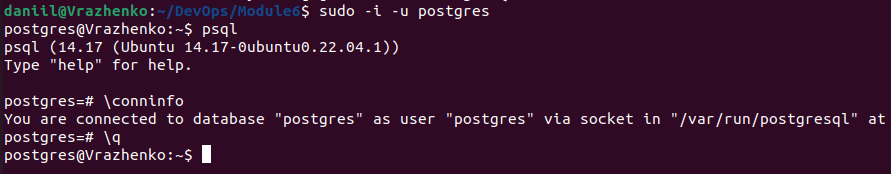
Установим postgresql 12 на ubuntu (рис. 1):

Рисунок 1 - Установка PostgreSQL

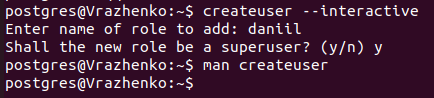
Проверим работоспособность базы данных (рис. 2):

Рисунок 2 - Проверка работоспособности

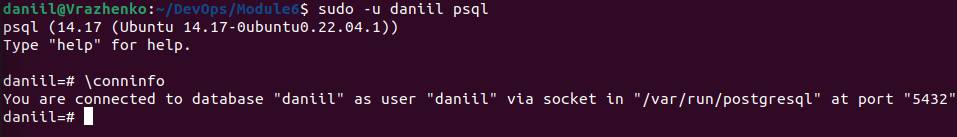
Войдём в potgres (рис. 3):

Рисунок 3 - Вход в postgres

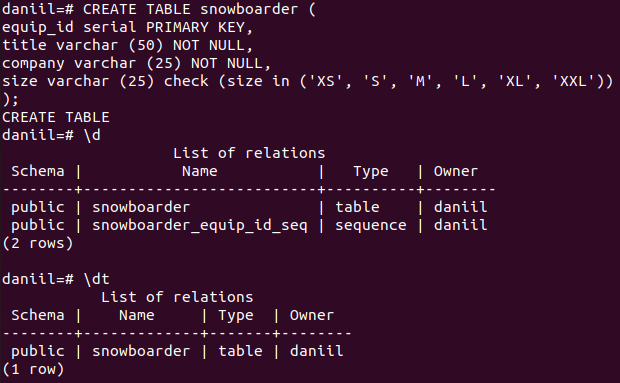
Создадим пользователя (рис. 4):

Рисунок 4 - Создание пользователя

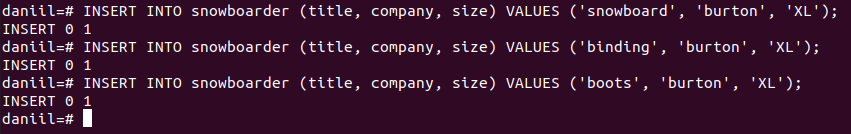
Создадим базу данных (рис. 5):

Рисунок 5 - Создание базы данных

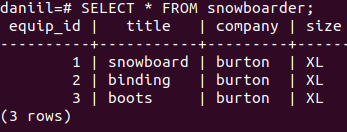
Создадим и просмотрим таблицу в базе данных (рис. 6):

Рисунок 6 - Создание и просмотр базы данных

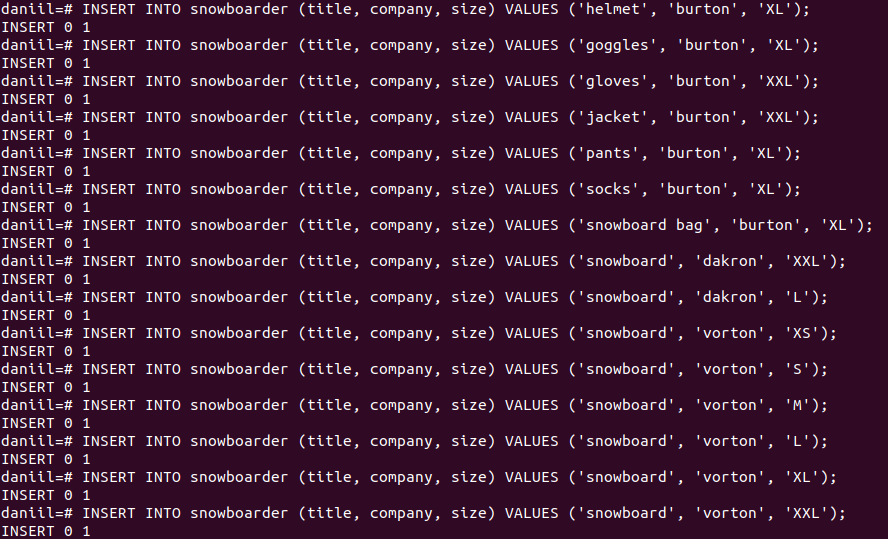
Заполним таблицу данными (рис. 7):

Рисунок 7 - Заполнение таблицы данными

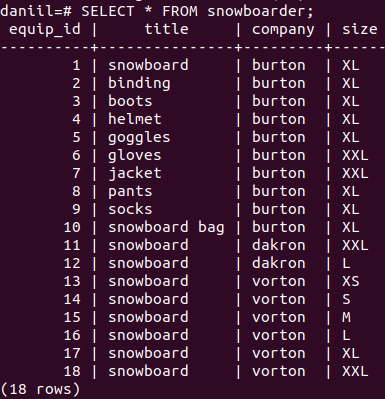
Проверим таблицу (рис. 8):

Рисунок 8 - Проверка таблицы

Дополним таблицу новыми данными (рис. 9):

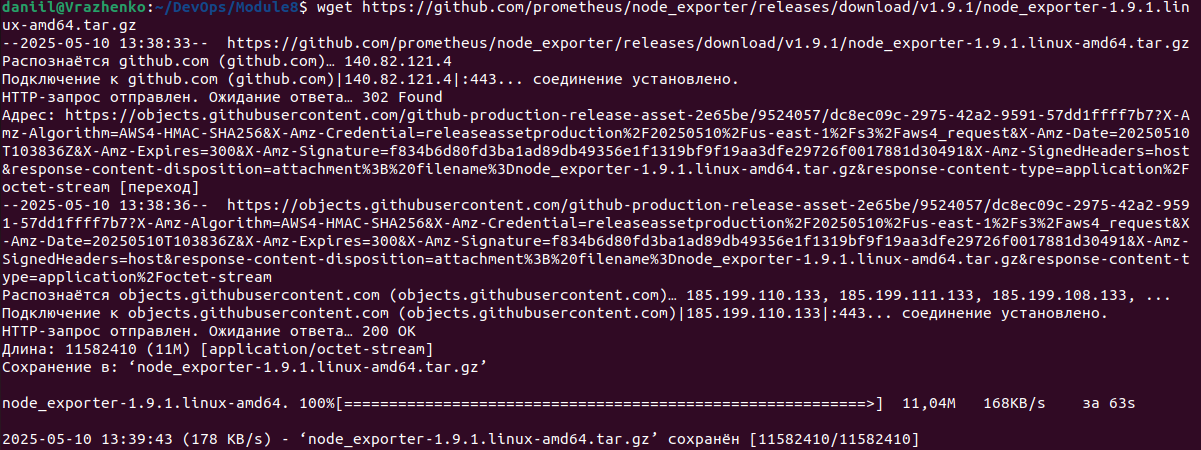
Рисунок 9 - Дополнение таблицы данными

Снова проверим таблицу (рис. 10):

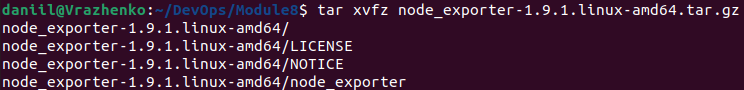
Рисунок 10 - Повторная проверка таблицы

# Модуль 8. Мониторинг, логирование и оповещение событий

Загрузим сжатый пакет node\_exporter (рис. 11):

Рисунок 11 - Загрузка node\_exporter

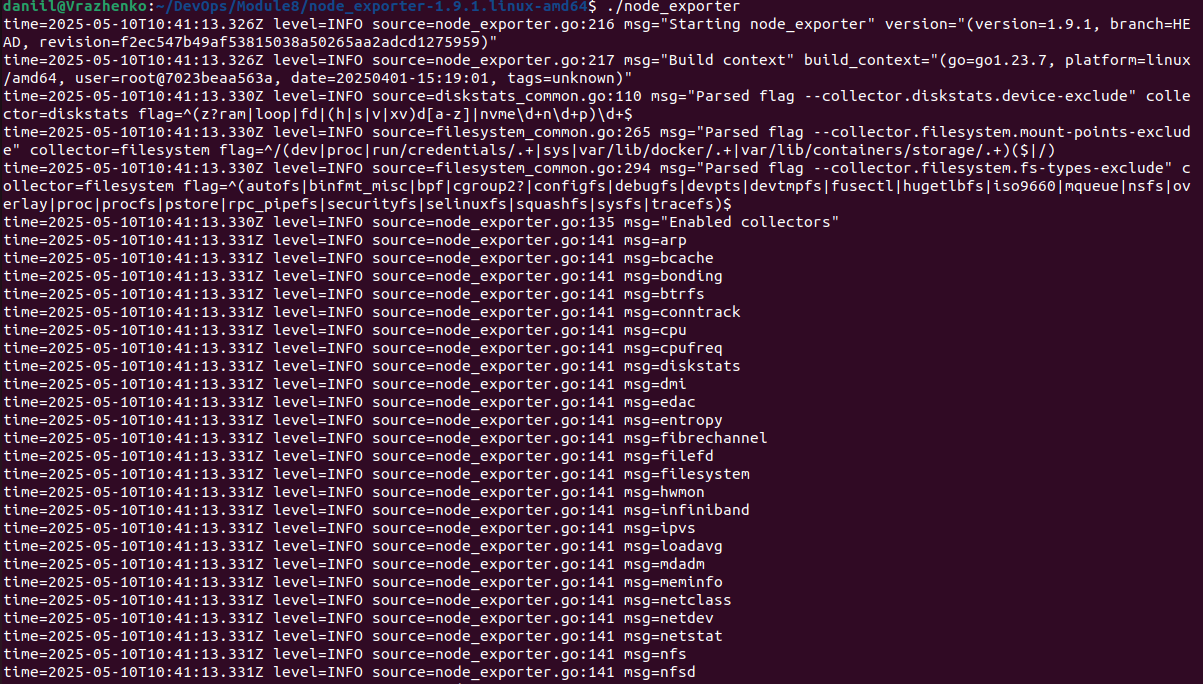
Извлечем двоичный файл node\_exporter (рис. 12):

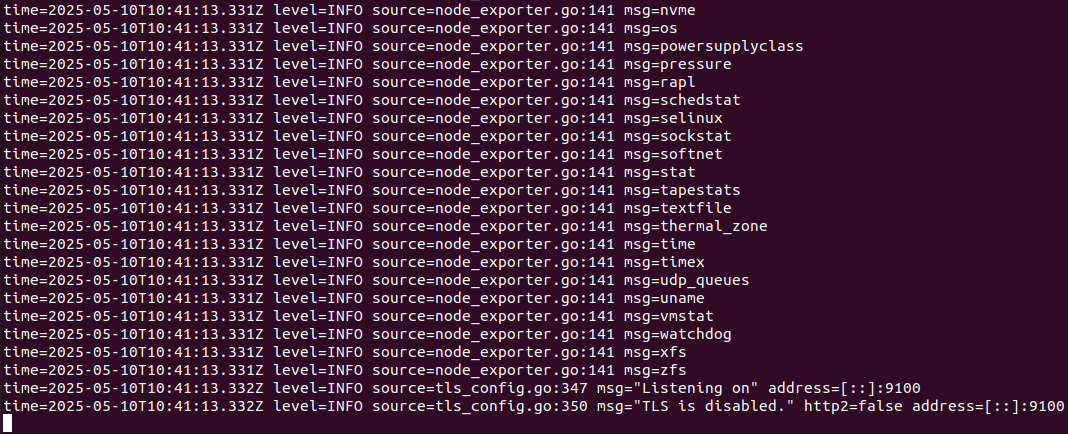
Рисунок 12 - Извлечение двоичного файла node\_exporter

Перейдем в каталог и сделаем двоичный файл исполняемым (рис. 13):

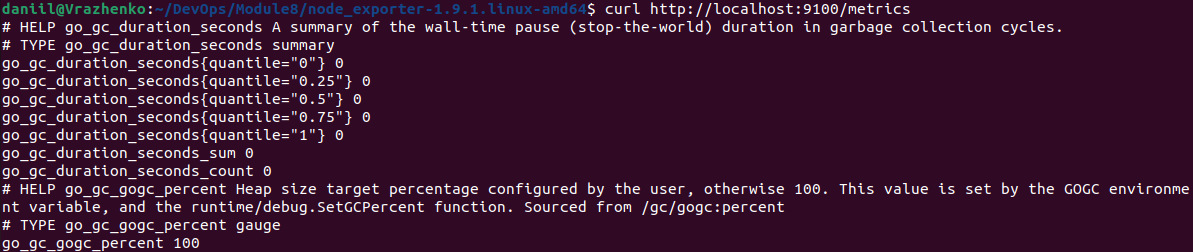
Рисунок 13 - Переход в каталог и изменение прав доступа

Запустим node\_exporter и увидим вывод, указывающий на то, что Node Exporter теперь запущен на порту 9100 (рис. 14-15)

Рисунок 14 - Первая часть вывода

Рисунок 15 - Вторая часть вывода

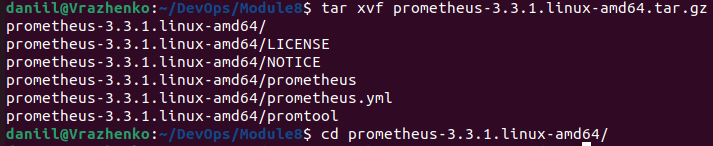
Проверим, что метрики экспортируются через порт 9100 (рис. 16):

Рисунок 16 - Наименьшая часть вывода

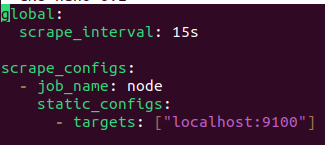
Загрузим архив пакета Prometheus (рис. 17):

Рисунок 17 - Загрузка пакета Prometheus

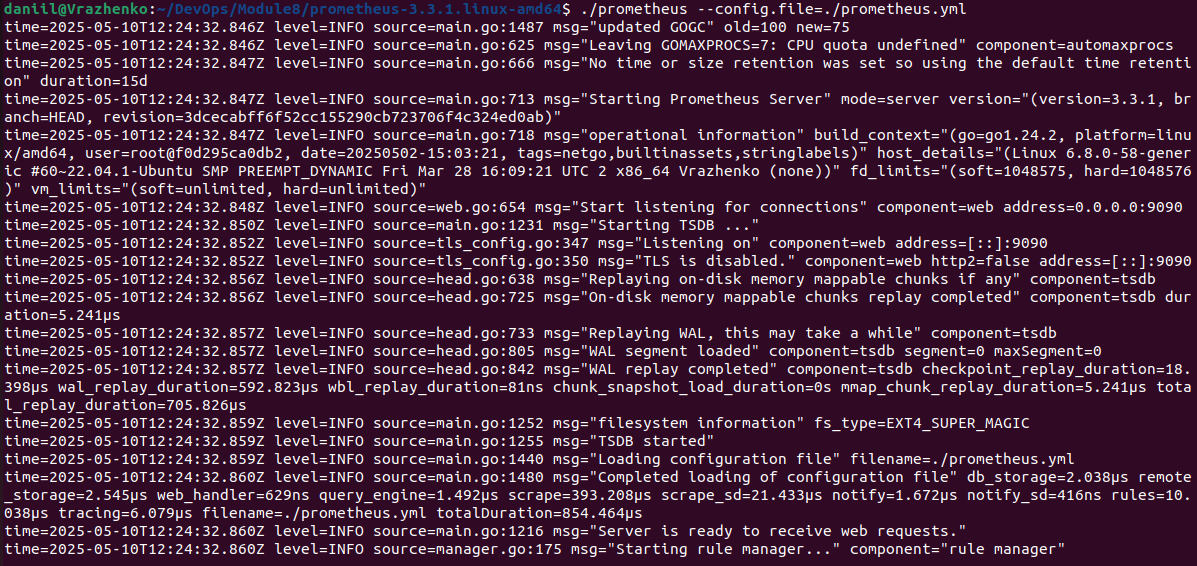
Извелечем двоичный файл и перейдем в каталог (рис. 18):

Рисунок 18 - Извлечение двоичного файла и переход в каталог

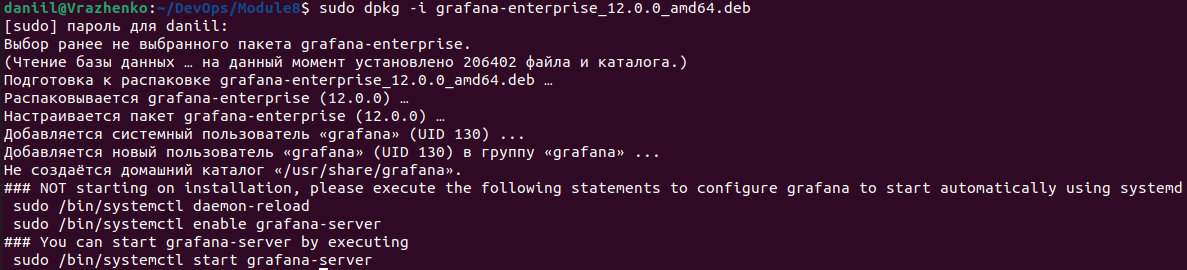
Настроим Prometheus с помощью изменения содержимого файла prometheus.yml (рис. 19):

Рисунок 19 - Содержимое prometheus.yml

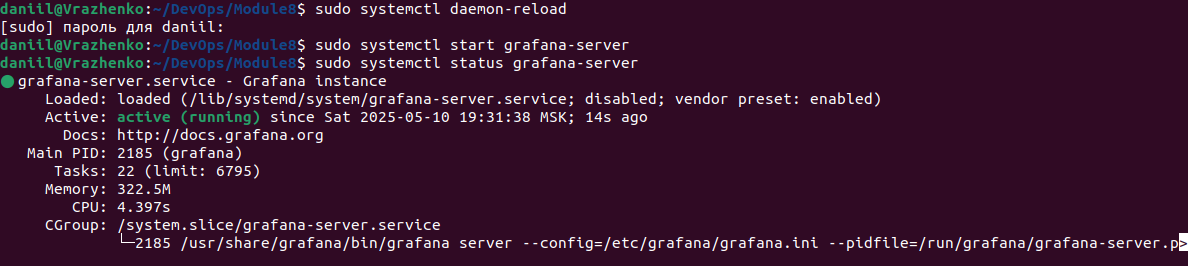
Запустим Prometheus (рис. 20):

Рисунок 20 - Запуск Prometheus

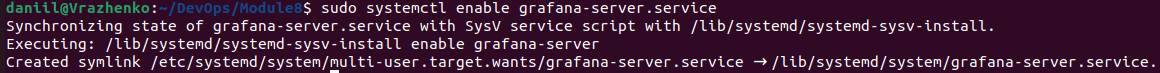
Выполним установку с помощью пакета .deb (рис. 21):

Рисунок 21 - Распаковка пакета .deb

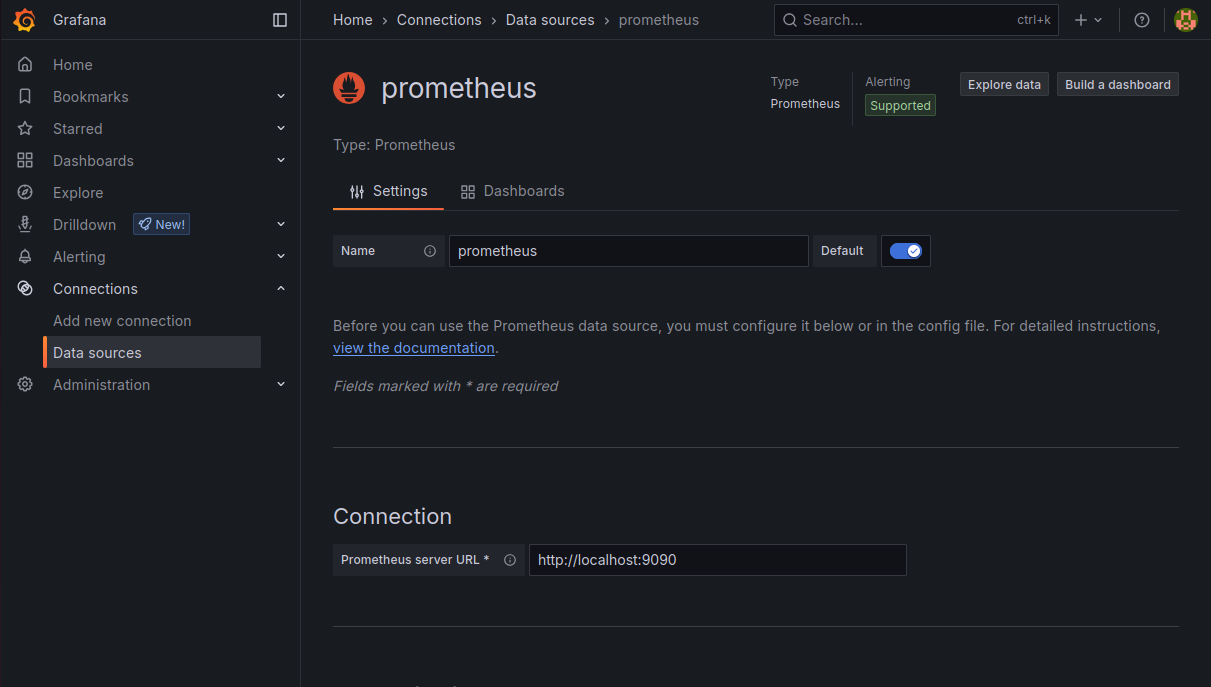
Запустим grafana-server (рис. 22):

Рисунок 22 - Запуск grafana-server

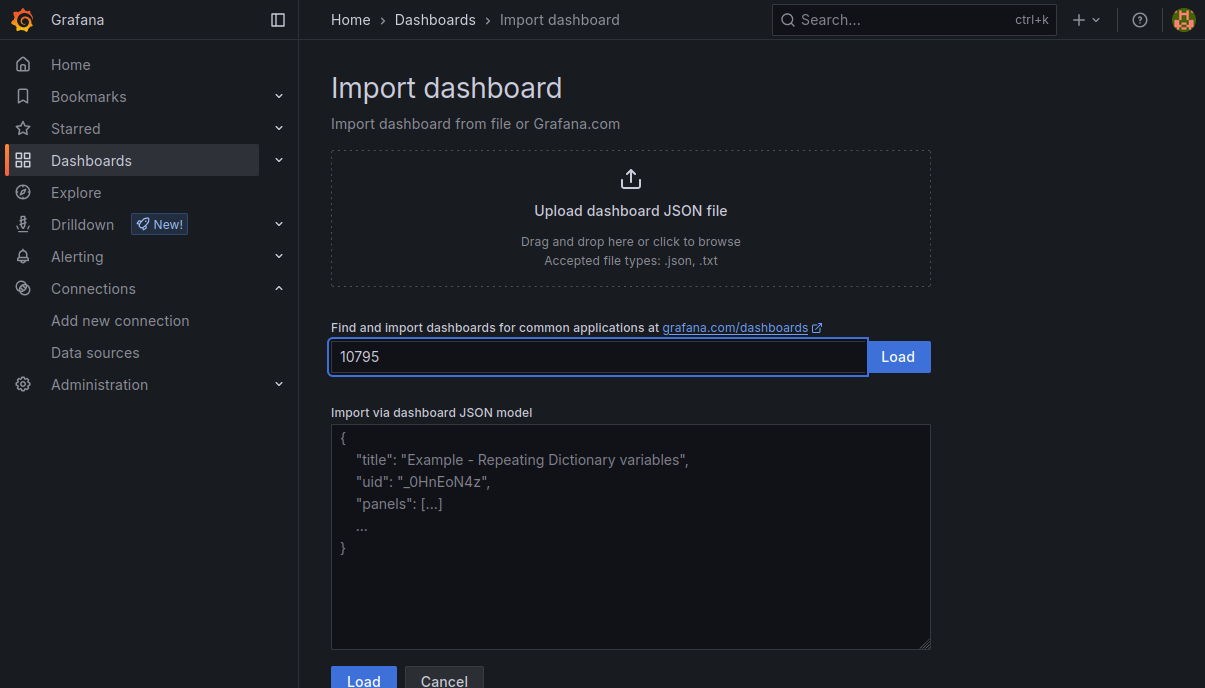
Настроим grafana-server для запуска при загрузке (рис. 23):

Рисунок 23 - Настройка grafana-server

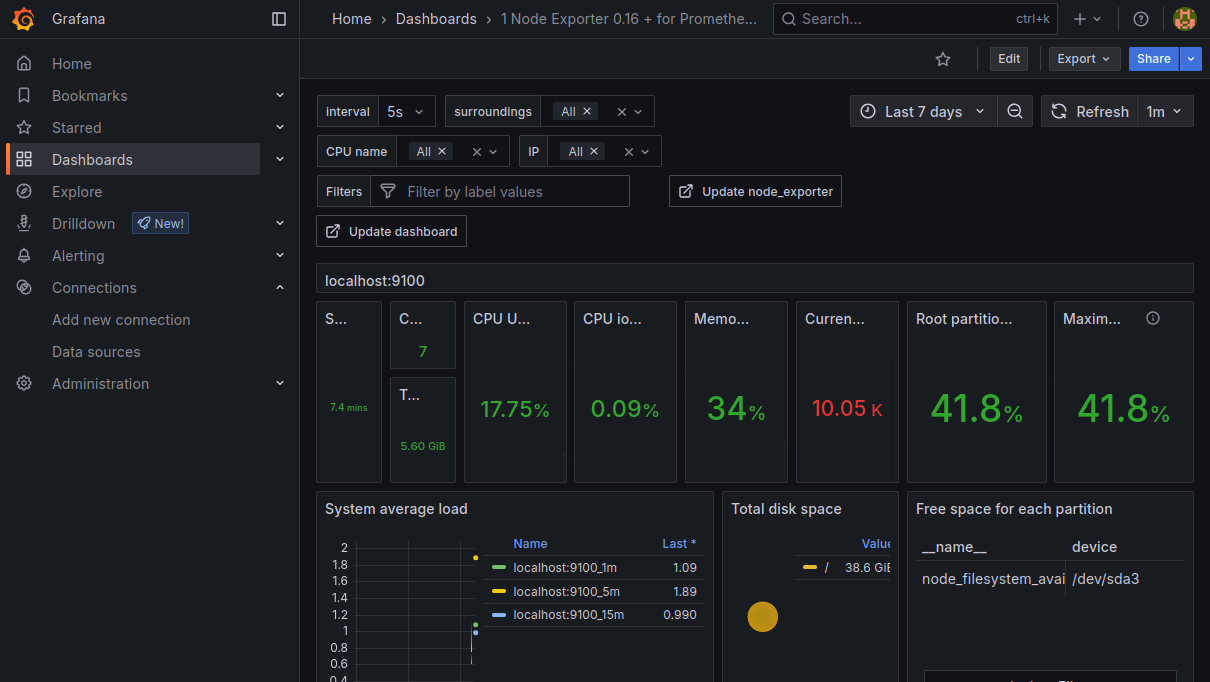
Настроим источники данных (рис. 24):

Рисунок 24 - Найстройки источников данных

Используем готовый дашборд (рис. 25):

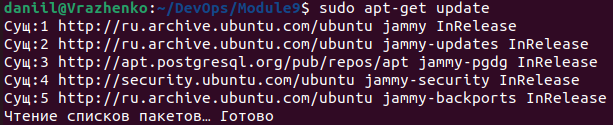
Рисунок 25 - Импорт готового дашборда

Откроем установленный дашборд (рис. 26):

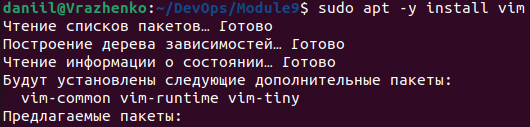
Рисунок 26 - Установленный дашборд

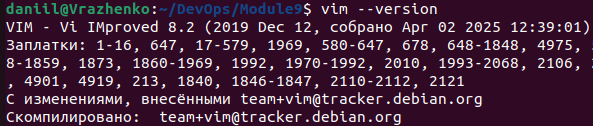
# Модуль 9. Виртуализация в DevOps

Обновим базу данных пакетов с помощью apt-get (рис. 27):

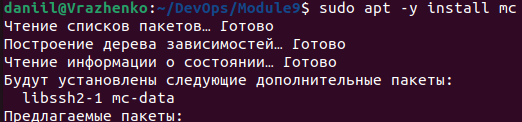
Рисунок 27 - Обновление базы данных пакетов

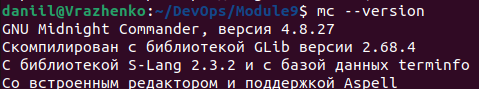
Установка vim и проверка версии (рис. 28-29):

Рисунок 28 - Установка vim

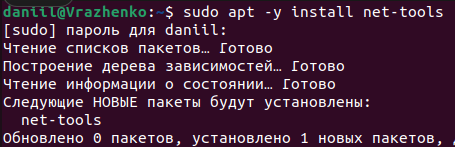
Рисунок 29 - Версия vim

Установка mc и проверка версии (рис. 30-31):

Рисунок 30 - Установка mc

Рисунок 31 - Версия mc

Установка net-tools (рис. 32):

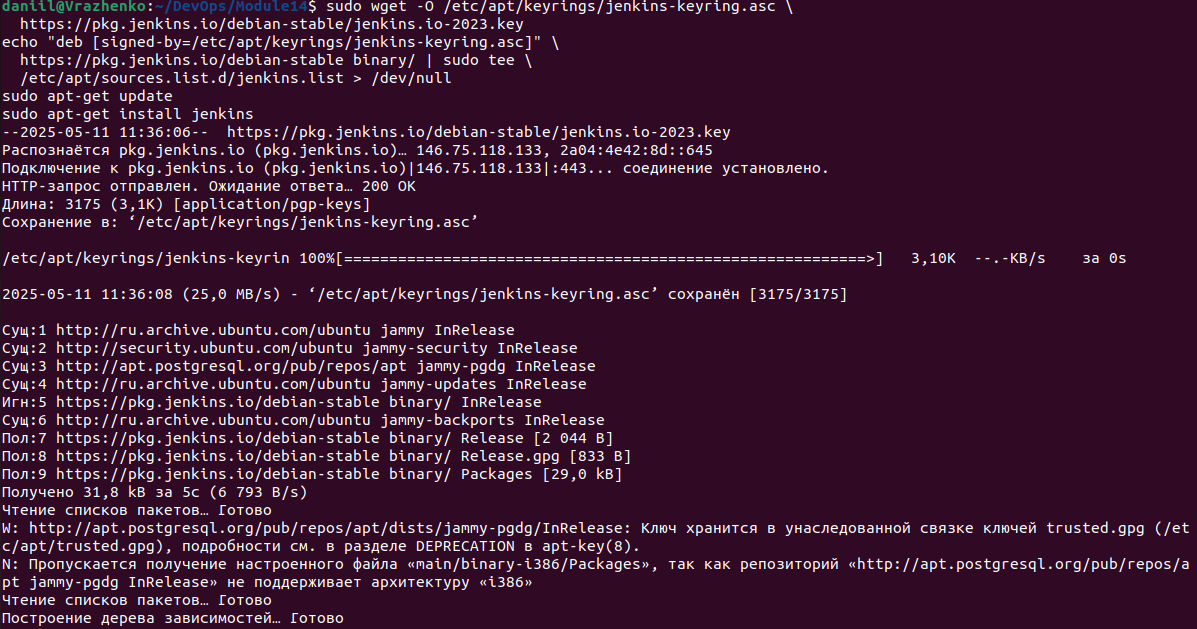
Рисунок 32 - Установка net-tools

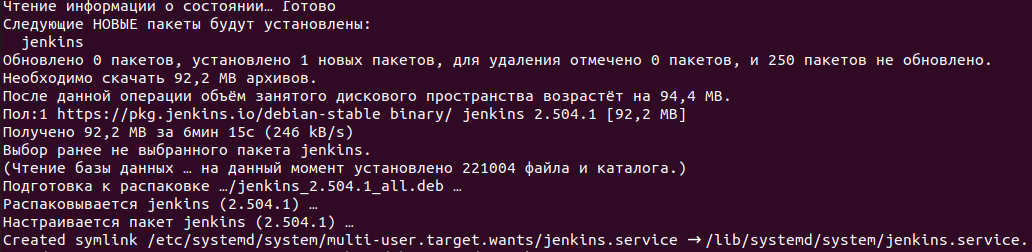
К сожалению, проверить версию net-tools напрямую (как, у vim или mc) не получится, так как net-tools — это набор отдельных утилит, а не единая программа. Однако можно использовать флаг --version с некоторыми инструментами из этого пакета, например, ifconfig (рис. 33):

Рисунок 33 - Версия net-tools

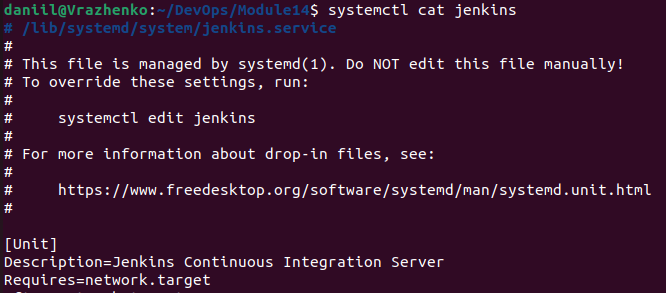
# Модуль 14. Жизненный цикл ПО

Установим Jenkins с помощью официальной инструкции (рис. 34-35):

Рисунок 34 - Первая часть установки

Рисунок 35 - Вторая часть установки

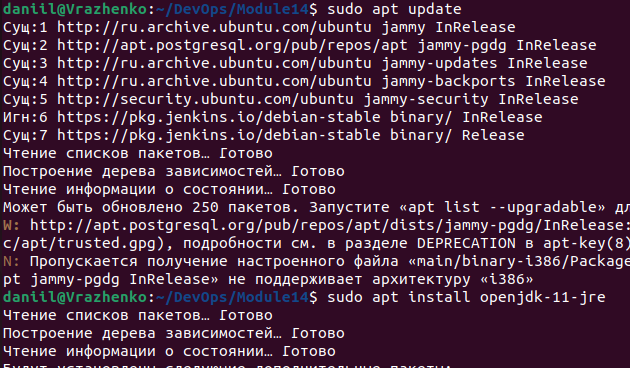
Настроим Jenkins как демон (рис. 36):

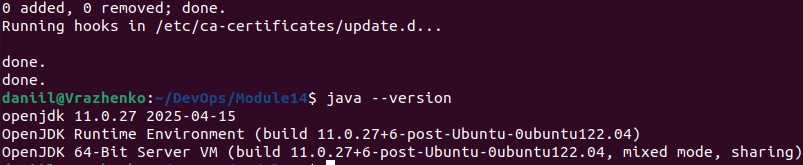
Рисунок 36 - Просмотр подробностей

Создадим пользователя для запуска службы (рис. 37):

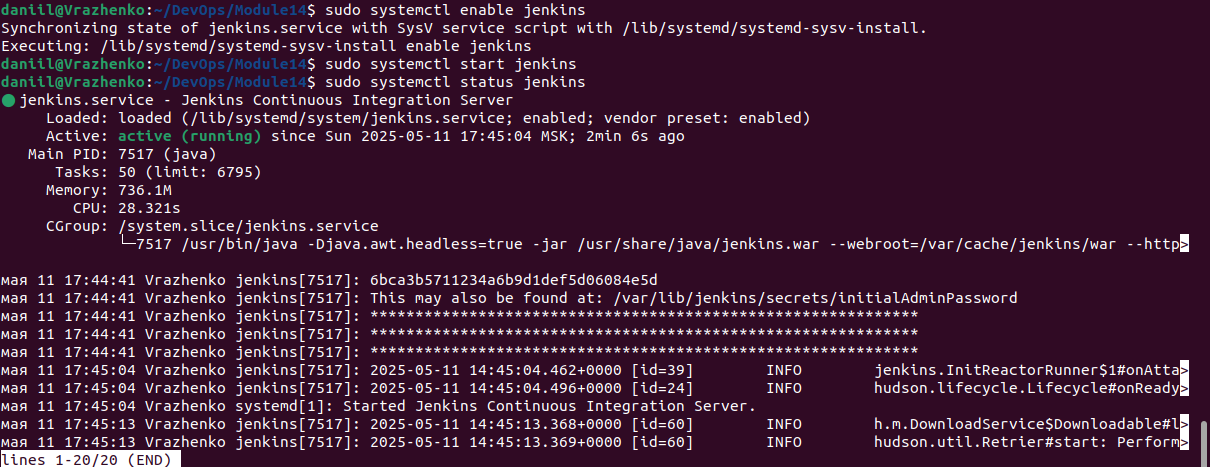
Рисунок 37 - id пользователя

Установим Java (рис. 38-39):

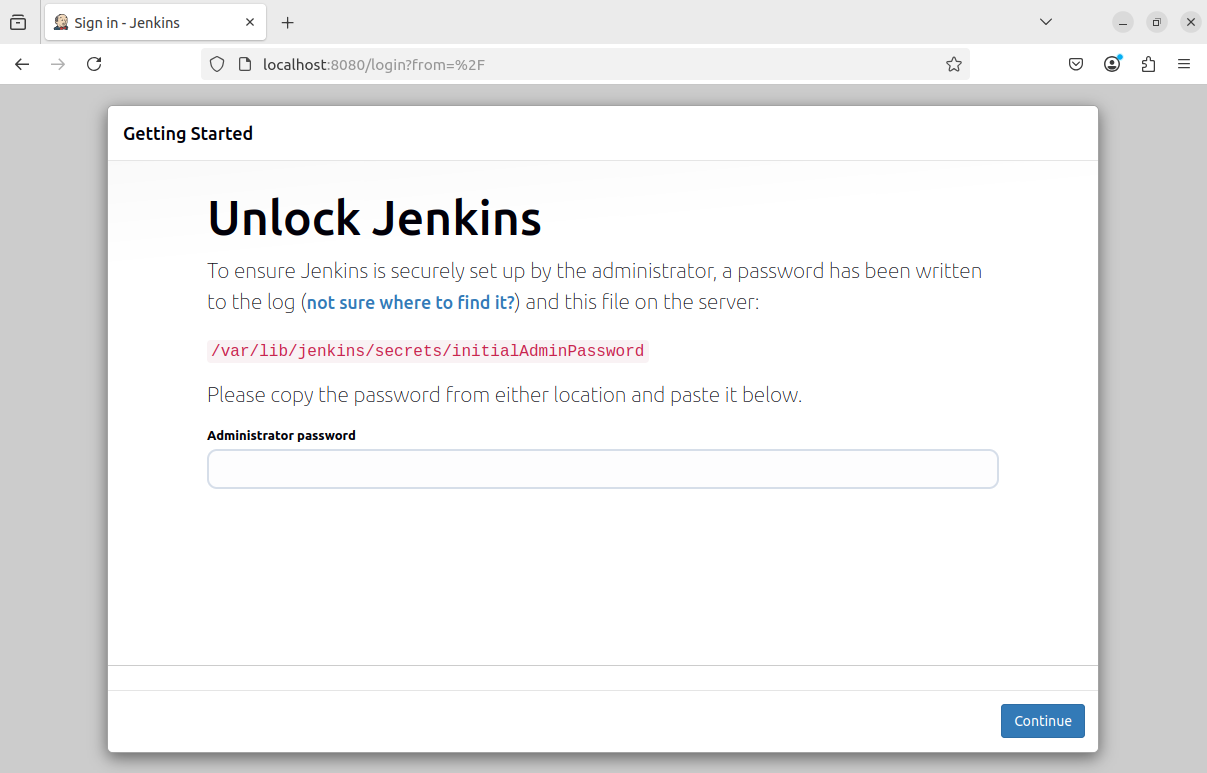
Рисунок 38 - Первая часть установки

Рисунок 39 - Вторая часть установки

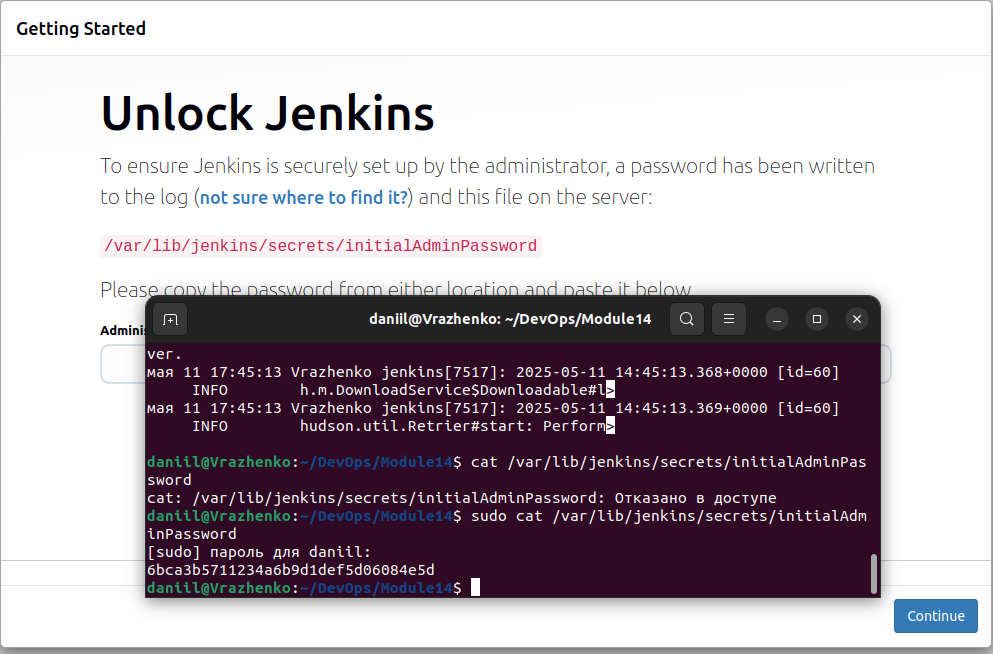
Запустим Jenkins (рис. 40):

Рисунок 40 - Запуск Jenkins и его статус

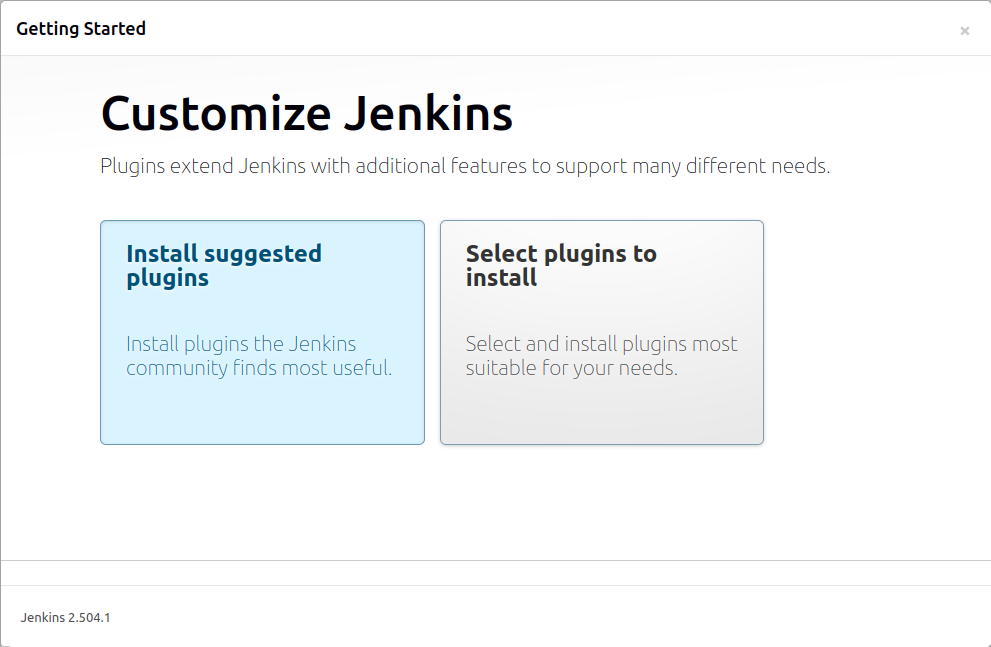
Разблокируем Jenkins (рис. 41):

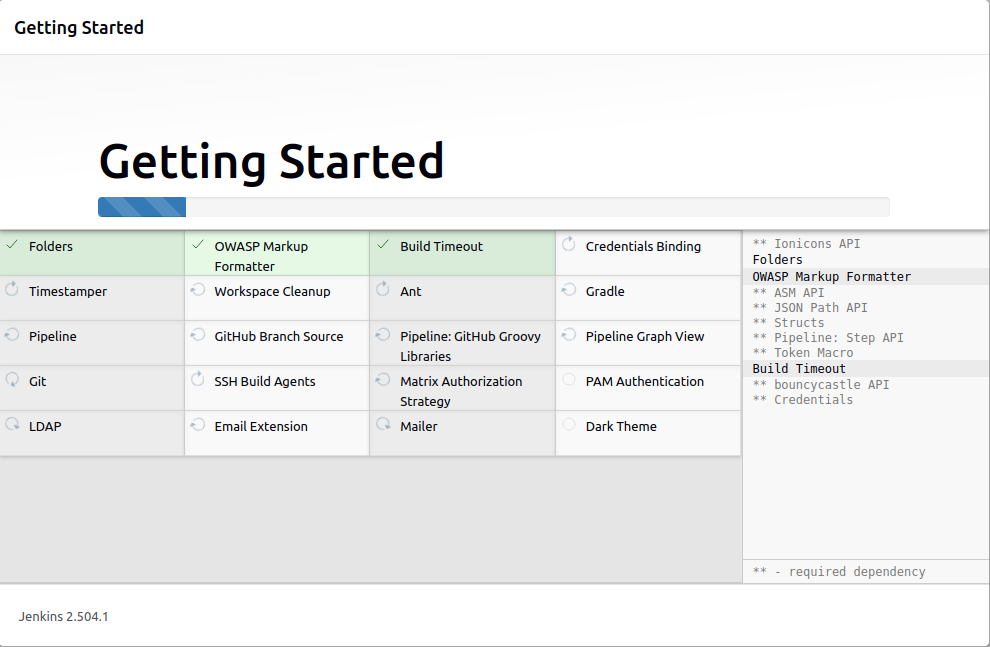
Рисунок 41 - Разблокировка Jenkins

Получим пароль для разблокировки Jenkins (рис. 42):

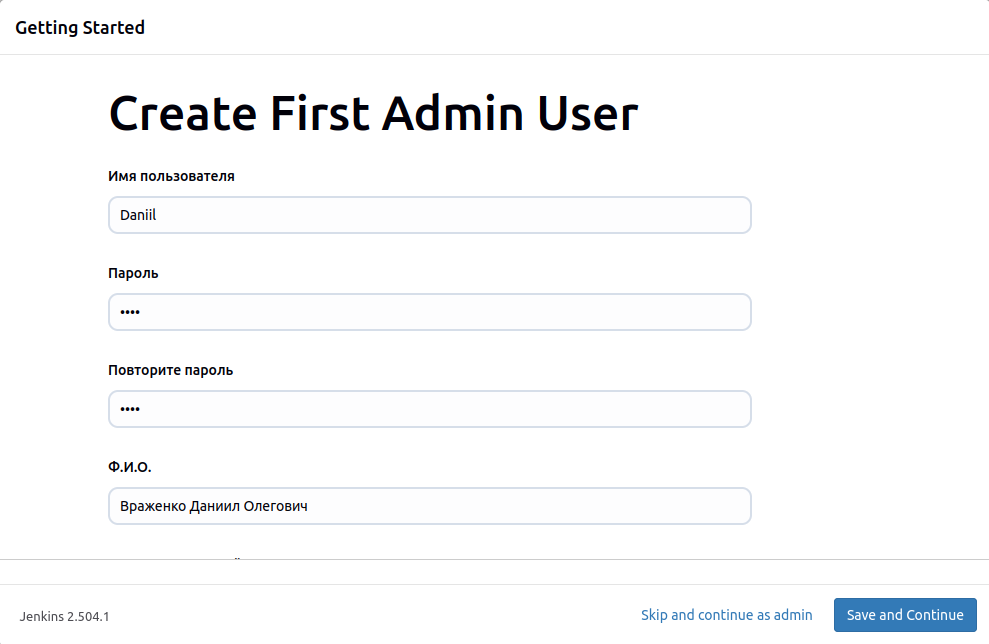
Рисунок 42 - Пароль для Jenkins

Установим плагины для Jenkins (рис. 43-44):

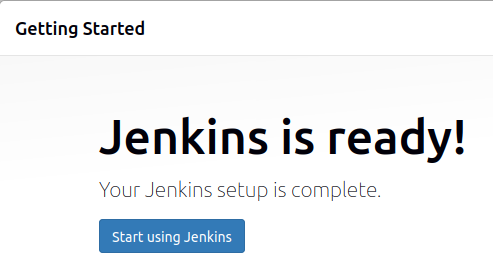
Рисунок 43 - Плагины для Jenkins

Рисунок 44 - Установка плагинов Jenkins

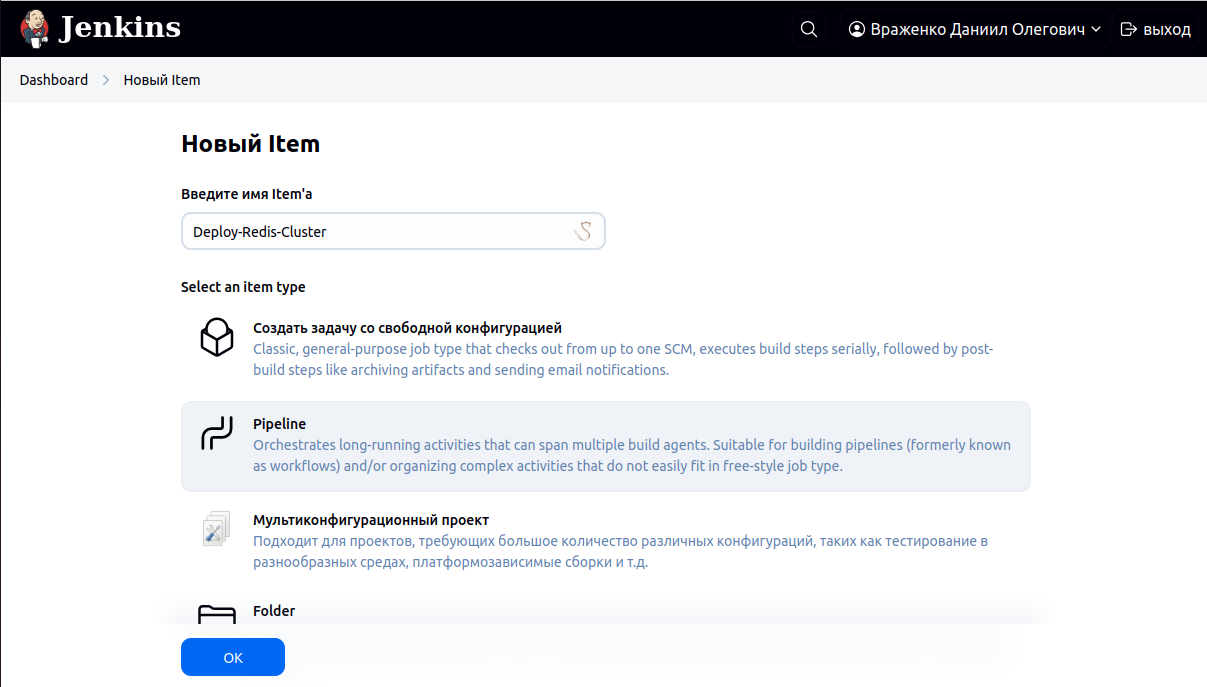
Создадим первого пользователя-администратора (рис. 45):

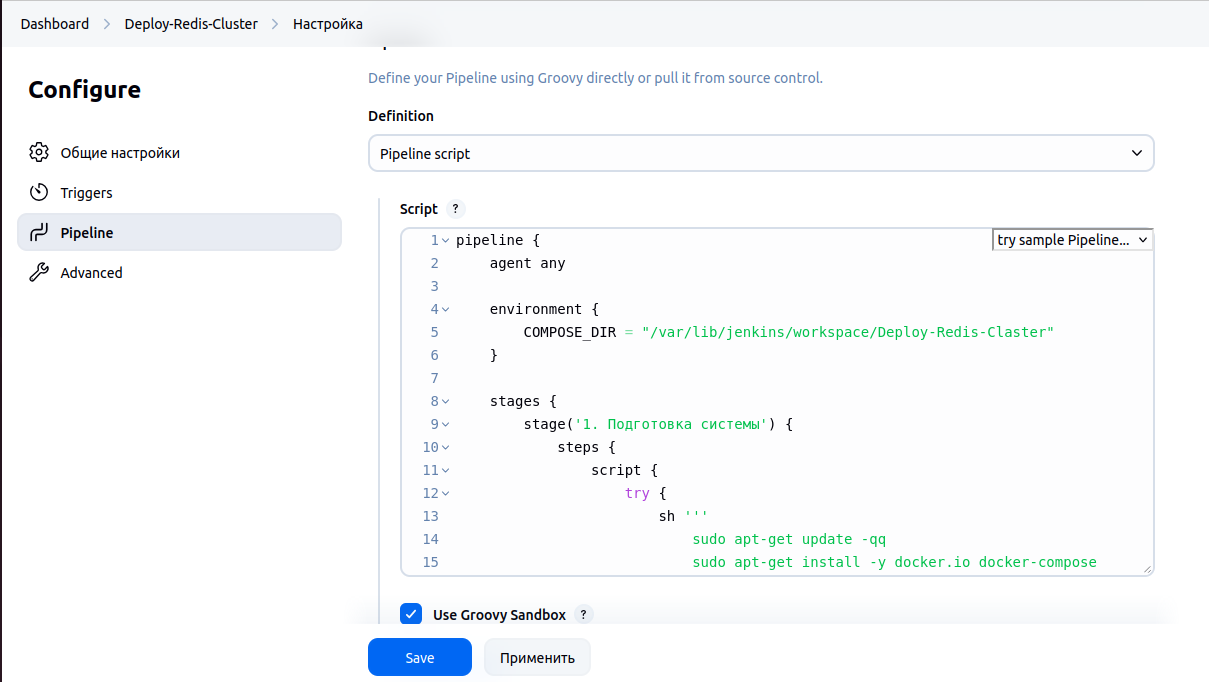
Рисунок 45 - Создание первого пользователя-администратора

Видим, что Jenkins готов к использованию (рис. 46-47):

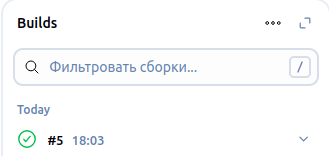
Рисунок 46 - Jenkins готов

Создадим Jenkins Pipeline и настроим его (рис. 47-48):

Рисунок 47 - Создание Pipeline

Рисунок 48 - Небольшая часть Pipeline скрипта

После нескольких попыток и изменений скрипта получаем успешный билд (рис. 49):

Рисунок 49 - Успешный билд

# Вывод

Мы проделали работу по следующим модулям:

**Модуль 6. Администрирование баз данных:**

* Установили PostgreSQL 12 на Ubuntu.
* Проверили работоспособность базы данных.
* Подключились к PostgreSQL через командную строку.
* Создали пользователя и базу данных.
* Сформировали таблицу, заполнили её данными и проверили корректность внесённых записей.
* Дополнили таблицу новыми данными и выполнили повторную проверку.

**Модуль 8. Мониторинг, логирование и оповещение событий:**

* Загрузили и распаковали Node Exporter, сделали его исполняемым и запустили на порту 9100.
* Убедились, что метрики успешно экспортируются.
* Установили Prometheus, настроили его конфигурационный файл prometheus.yml и запустили сервер.
* Развернули Grafana с помощью пакета .deb, настроили автозапуск сервера.
* Добавили Prometheus в качестве источника данных в Grafana.
* Импортировали готовый дашборд и проверили отображение метрик.

**Модуль 9. Виртуализация в DevOps:**

* Обновили базу пакетов Ubuntu.
* Установили и проверили версии утилит:
  + Vim
  + Midnight Commander (mc)
  + Net-tools

**Модуль 14. Жизненный цикл ПО:**

* Установили Jenkins, следуя официальной инструкции.
* Настроили Jenkins как демона, создали системного пользователя для его работы.
* Установили Java.
* Запустили Jenkins.
* Установили рекомендуемые плагины.
* Создали административного пользователя и завершили первоначальную настройку.