# 

# Выполнение практической работы

**Цель работы:** получить навыки настройки вычислительной инфраструктуры при помощи системы конфигурационного управления Ansible.

## Подготовка инфраструктуры

Прежде всего, необходимо создать 2 виртуальные машины Ubuntu или Debian. Этими машинами необходимо будет управлять в ходе практики. Имя пользователя должно быть указано как user. Это важно для дальнейшей работы.

После заполнения перезапустите машину командой reboot. Ещё одним важным шагом является установка ssh-сервера, который позволит удалённо подключаться к машине. Установить его можно при помощи команды:

*apt install ssh*

Также необходимо создать управляющую машину, на которой непосредственно будет установлен Ansible. Это может быть как виртуальная машина, тогда необходимо будет проделать те же самые манипуляции, что и для управляемых машин, так и хостовая машина. Для пользователей Windows рекомендуется воспользоваться WSL2, который позволяет поднять Linux прямо на хостовой машине. Работа проводится на Windows, WSL 2, Ubuntu.

Первый эмулированный пользователь находится на 22 порту, fed1, рис 1.

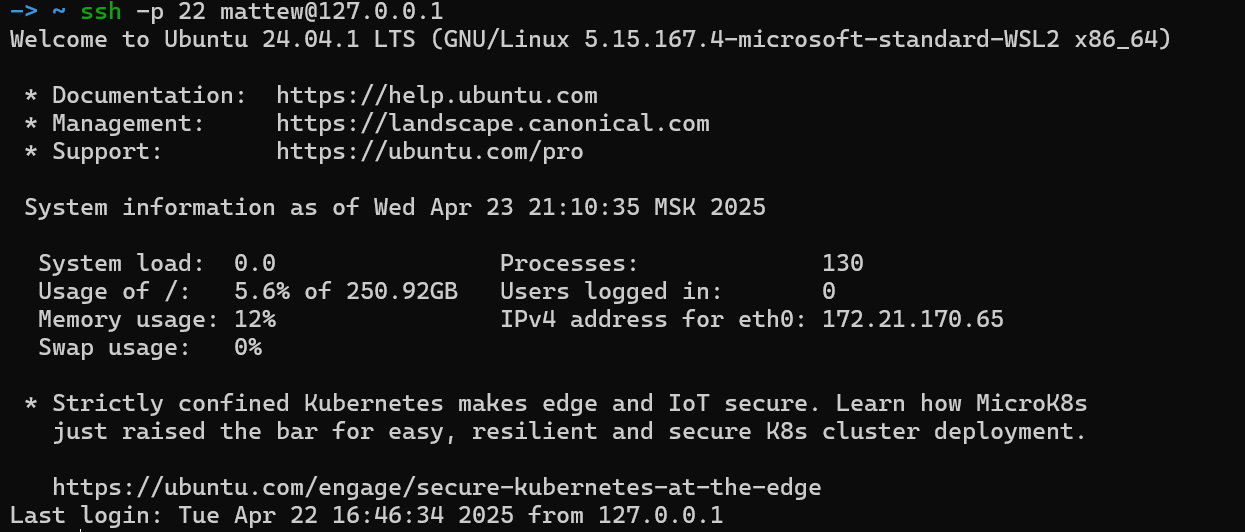
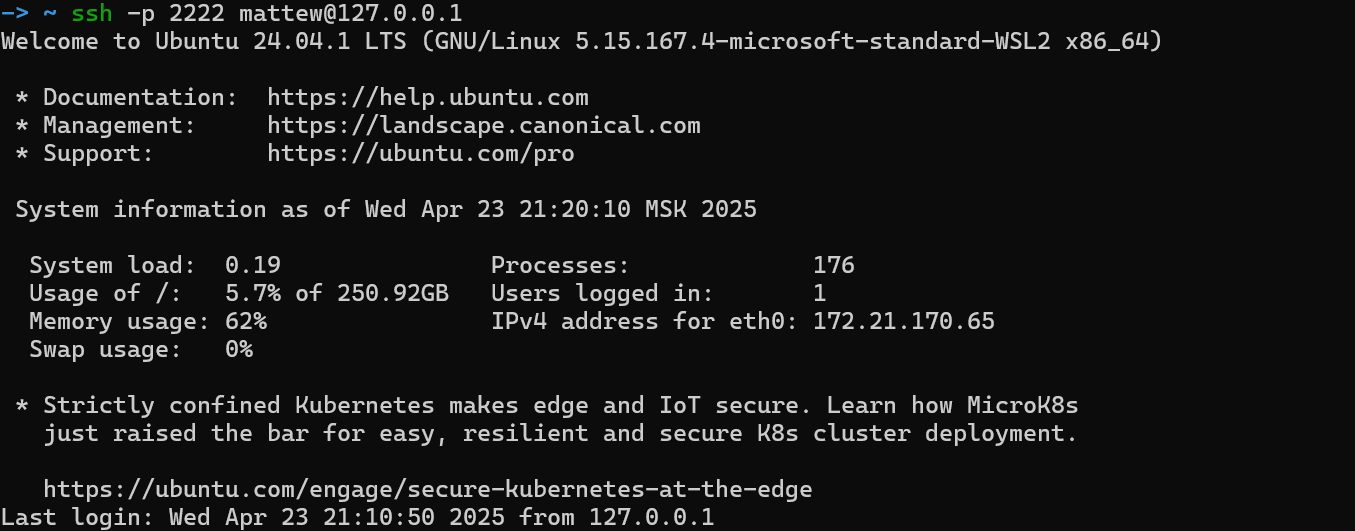


Рисунок 1 – Первая виртуальная машина, подключение 22 порт

Второй эмулированный пользователь находится на 2222, fed2, рис 2.

Рисунок 2 – Вторая виртуальная машина, подключение 2222 порт

## Установка Ansible

Для начала необходимо установить сам Ansible на управляющую машину. Сделать это можно при помощи следующих команд: sudo apt install ansible - будет установлена версия Ansible, содержащаяся в текущем выпуске дистрибутива sudo pip3 install ansible - установка при помощи пакетного менеджера Python 3 (поскольку Ansible написан на Python).

Для установки через pip может потребоваться его установить. Сделать это можно командой sudo apt install python3-pip. Наличие Ansible можно проверить командой ansible --version. После чего попробуем подключиться к управляемым машинам при помощи ssh. ssh root@ip\_address где ip\_address - то адрес управляемого узла в сети, рис 3.



Рисунок 3 – Отладка работы ansible на хостовой машине

## Настройка Ansible

Все действия также необходимо выполнять из-под пользователя root. Настроим inventory-файл. Создадим директорию ansible в домашнем каталоге пользователя для хранения репозитория инфраструктуры, будем считать эту директорию рабочей, и в ней создадим файл hosts. Заполните файл по следующему образу. Inventory-файл содержит в себе всю необходимую информацию о хостах, которые должны управляться системой Ansible. Выполним самую первую команду для проверки работы Ansible:

*ansible -i ./hosts -m ping all*

Данная команда выполнит команду ping для всех хостов в inventory и выдаст результат выполнения, рис 4.

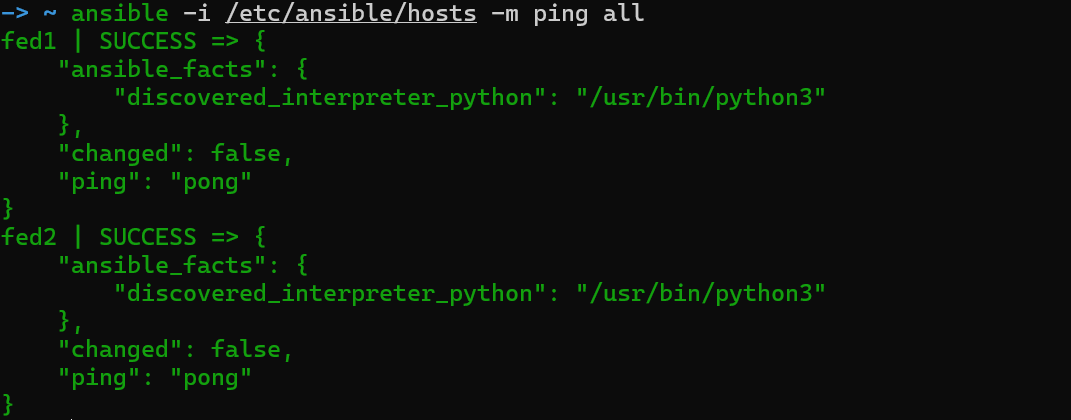


Рисунок 4 – Проверка подключения к виртуальным машинам

## Использование Ansible для конфигурации хостов

**Установка Ansible**

Ansible — это инструмент автоматизации без агентов, который вы устанавливаете на одном хосте (называемом узлом управления). С управляющего узла Ansible может удаленно управлять целым парком машин и других устройств (называемых управляемыми узлами) с помощью SSH, удаленного управления Powershell и множества других транспортных средств, и все это с помощью простого интерфейса командной строки, не требующего баз данных или демонов.

**Факты**

Факты — это параметры, которыми можно управлять в реализуемых сценариях.

Соберем возможные факты с управляемого хоста fed1, рис. 5:

*ansible fed1 -i ./hosts -m command -m setup*

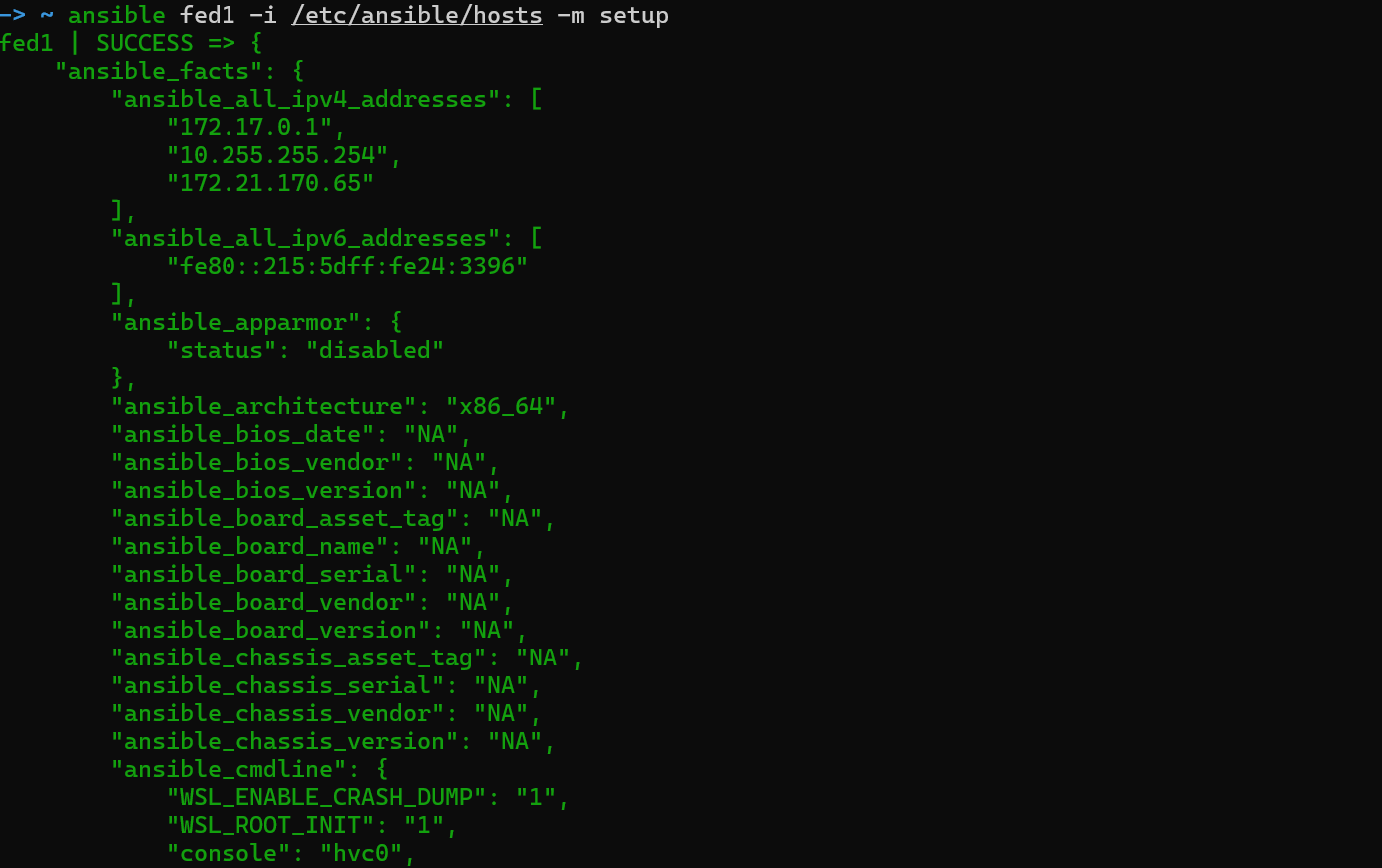
**

Рисунок 5 – Факты машины fed1

**Playbook**

Это конфигурационный сценарии, написанный на языке YAML, который впоследствии будет выполнятся на управляемых хостах.

Напишем playbook, который установит веб-сервер Nginx на управляемые хосты.

Сам файл с playbook’OM состоит из сценариев. Каждый сценарий начинается с ключевого слова name.

tasks — это список задач, которые необходимо выполнить на управляемой машине в ходе сценария. Каждый элемент списка задач начинается с символа дефис (-). Этот символ в языке YAML обозначает начало ассоциативного массива.

При помощи ключевого слова name задается имя задачи. При помощи ключевого слова when задаются условия для запуска задачи, к примеру в указанном примере задаётся условие, что задача по установке nginx должна выполнятся только на хосте под управлением Debian.

handlers — это действия, которые будут выполняться после завершения задачи. Они запускаются на выполнения только 1 раз и после завершения всего сценария. Ключевое слово notify позволяет запускать handler’bi. В указанном примере после установки nginx будет запущен handler, активирующий автозапуск сервиса nginx.

После написания р1ауbоок’а его можно выполнить при помощи следующей команды ansible-playbook -i hosts

В результате успешного запуска вы должны увидеть следующий вывод в терминале, рис 6.

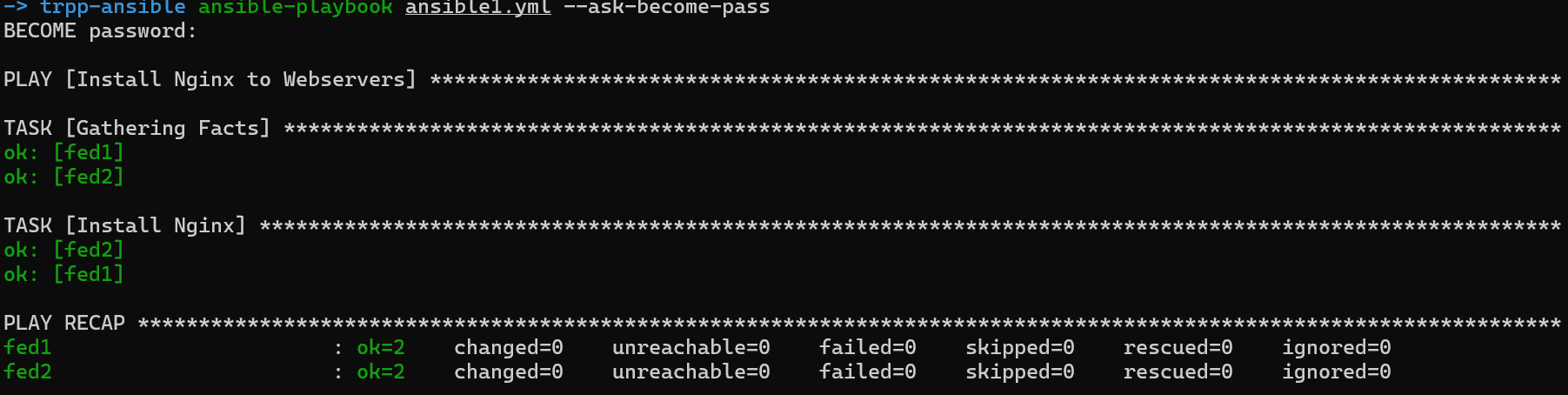
****

Рисунок 6 – Результат выполнения рlауbook’а

После этого можно перейти на любой из двух узлов (при помощи ssh или же непосредственно открыв окно виртуальной машины) и убедиться в том, что nginx действительно установлен при помощи команды systemctl status nginx.

Теперь остановим и удалим с данного узла nginx при помощи следующих команд непосредственно через терминал.

*systemctl stop nginx*

*apt remove nginx*

Теперь повторно необходимо запустить playbook, рис 7

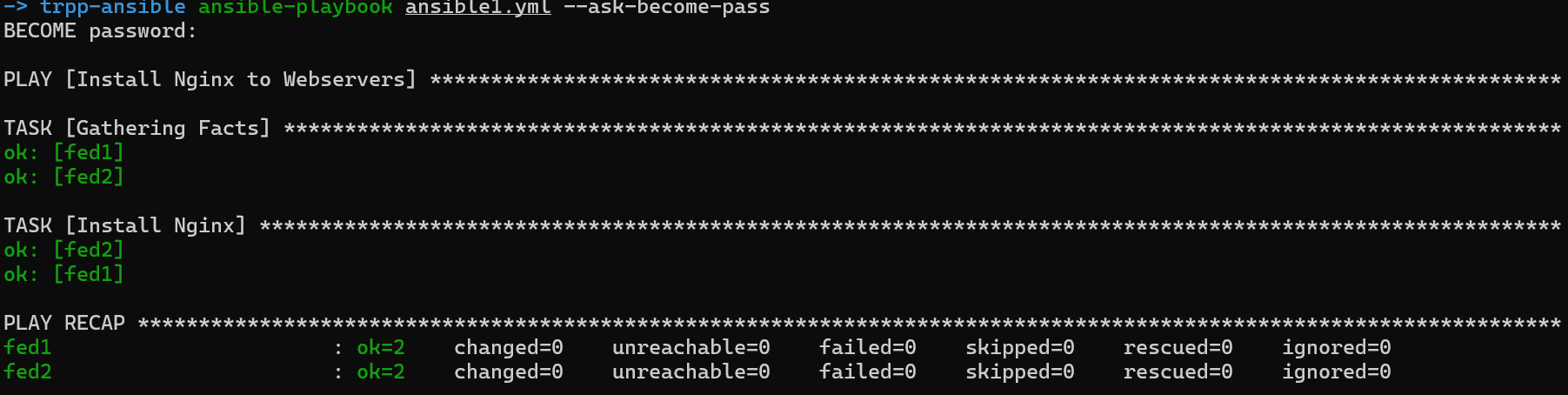
****

Рисунок 7 – Результат повторного выполнения рlауbook’а

**Более сложный playbook**

Начнем с написания более сложного playbook’a и рассмотрим отдельные его составляющие:

*---*

*- name: Install and config Nginx*

*hosts: all*

*become: yes*

*vars:*

*html\_dir: /usr/share/nginx/html*

*greeting: "Hello Everybody!"*

*tasks:*

*- name: Install Nginx*

*apt:*

*name: nginx*

*state: present*

*update\_cache: yes*

*notify:*

*- Nginx Systemd*

*- name: Delete default HTML files*

*shell: /bin/rm -rf /usr/share/nginx/html/\*.html*

*- name: Replace config file*

*vars:*

*nginx\_user: user*

*worker\_processes: 2*

*worker\_connections: 256*

*template:*

*src: templates/nginx.conf.j2*

*dest: /etc/nginx/nginx.conf*

*mode: 0644*

*register: result*

*failed\_when: result.failed == true*

*notify: Reload Nginx*

*- name: Copy index file*

*copy:*

*src: files/index.html*

*dest: {{ html\_dir }}*

*mode: 0644*

*notify: Reload Nginx*

*- name: Generate dynamic HTML from template*

*template:*

*src: templates/hello.html.j2*

*dest: {{ html\_dir }}/hello.html*

*owner: root*

*mode: 0644*

*notify: Reload Nginx*

*handlers:*

*- name: Nginx Systemd*

*systemd:*

*name: nginx*

*enabled: yes*

*state: started*

*- name: Reload Nginx*

*systemd:*

*name: nginx*

*state: reloaded*

В разделе vars описываются переменные, которые затем можно использовать при запуске задач. Для использования переменных в задачах используется синтаксис шаблонизатора - {{ имя\_переменной }}.

Ещё одним инструментом является использование шаблонных файлов. Для этого используется шаблонизатор Jinja2. Шаблонные файлы добавляются в директорию templates с дополнительных расширением j2, то есть итоговое имя файла должно выглядеть, например, следующим образом index.html.j2. При реализации вышеописанного playbook’a используется 2 шаблонных файла: nginx.conf.j2, hello.html.j2.

Содержимое файла hello.html.j2, рис 8.

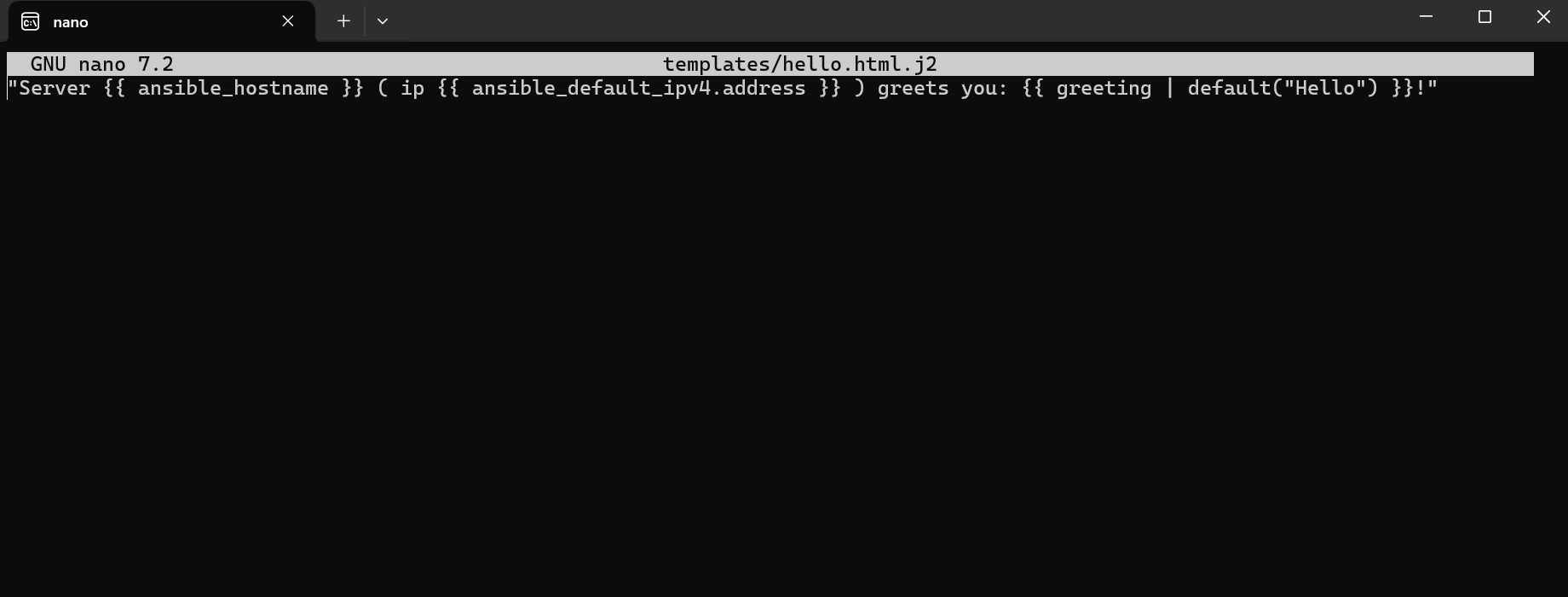


Рисунок 8 – Файл hello.html.j2

Содержимое файла nginx.conf.j2, рис 9.

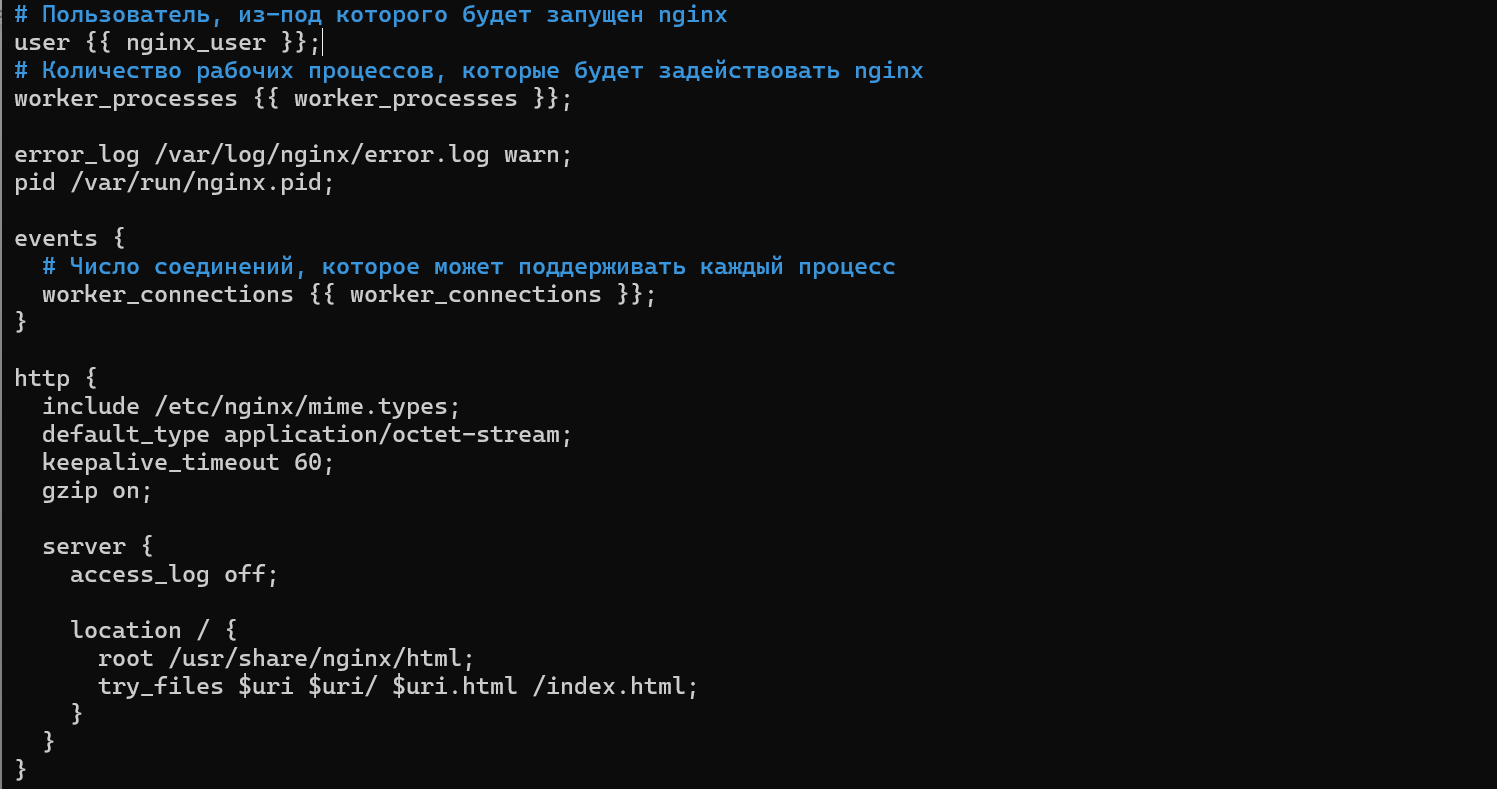


Рисунок 9 – Файл nginx.conf.j2

В результате содержимое директории должно иметь следующий вид, рис 10.

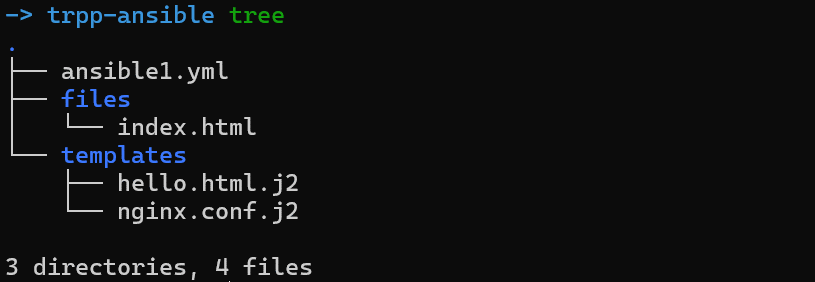


Рисунок 10 – Содержимое директории проекта

Теперь выполним команду для запуска playbook’a с пробным прогоном, который позволит проверить корректность написанного playbook’а без внесения изменений на целевые узлы, рис 11

*ansible-playbook -i hosts --check*

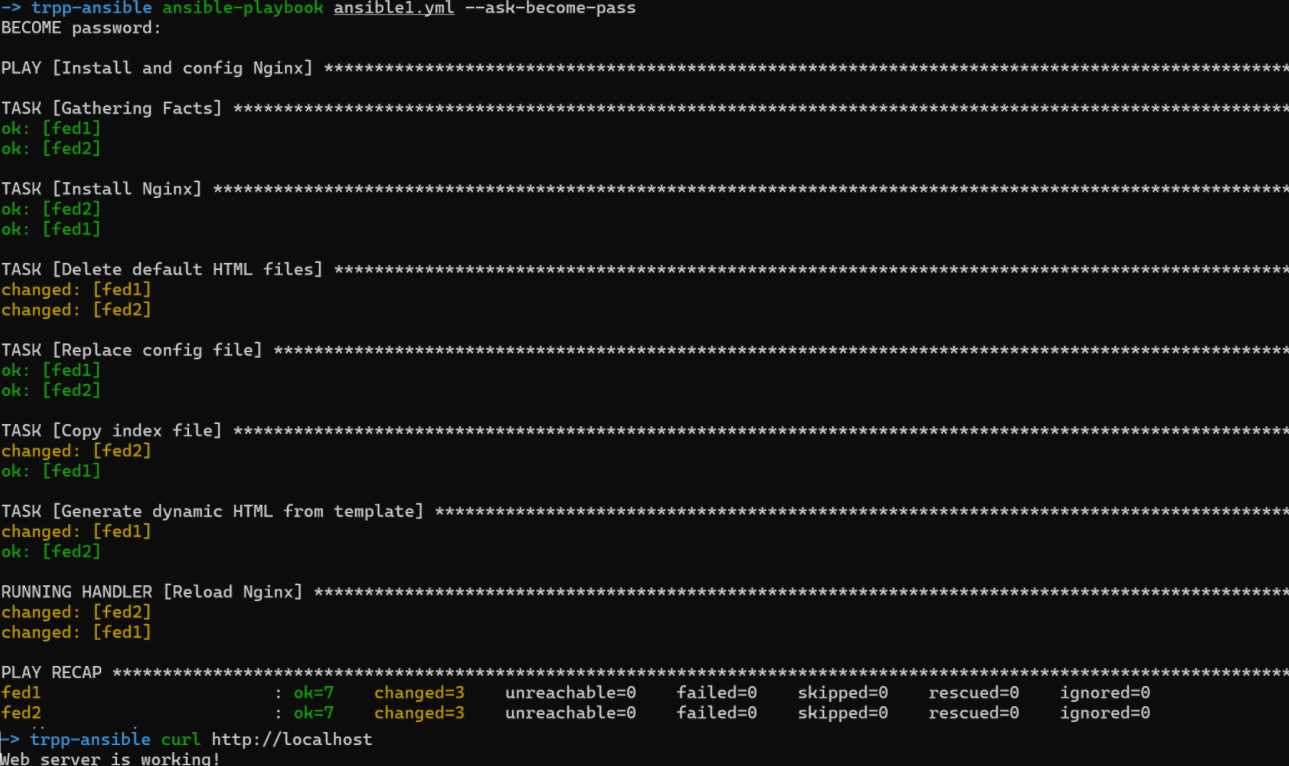


Рисунок 11 – Результат выполнения плейбука

Теперь внесём изменения в конфигурацию первого сервера, убрав ключ --check из предыдущей команды, а затем запросим базовую страницу при помощи всё той же утилиты curl.

*curl -L http://127.0.0.1*

Также можно запросить созданную при помощи шаблона страницу hello.

*curl -L http://127.0.0.1/hello*

Как можно увидеть, рис 13, изменения успешно применились. Можете самостоятельно поменять переменную greetings, затем загрузить повторно playbook на машину и понаблюдать за изменениями.

После этого применим playbook для всех серверов в группе, рис 12.

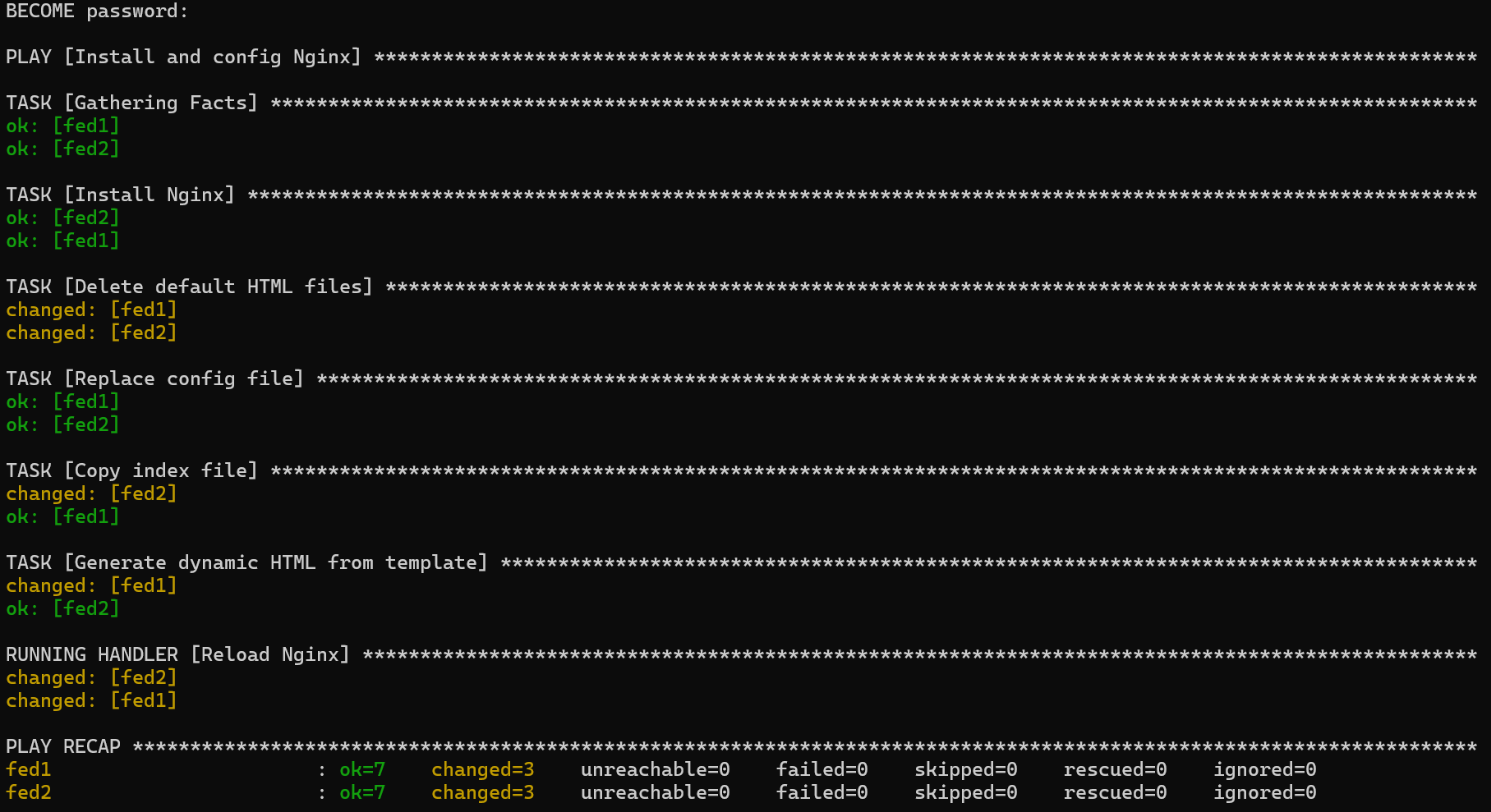


Рисунок 12 – Проверка работы плейбука

Как можно заметить, при небольшом расширении задачи, решаемой при помощи Ansible, файл playbook’a начинает разрастаться, что вызывает неудобство при его чтении.

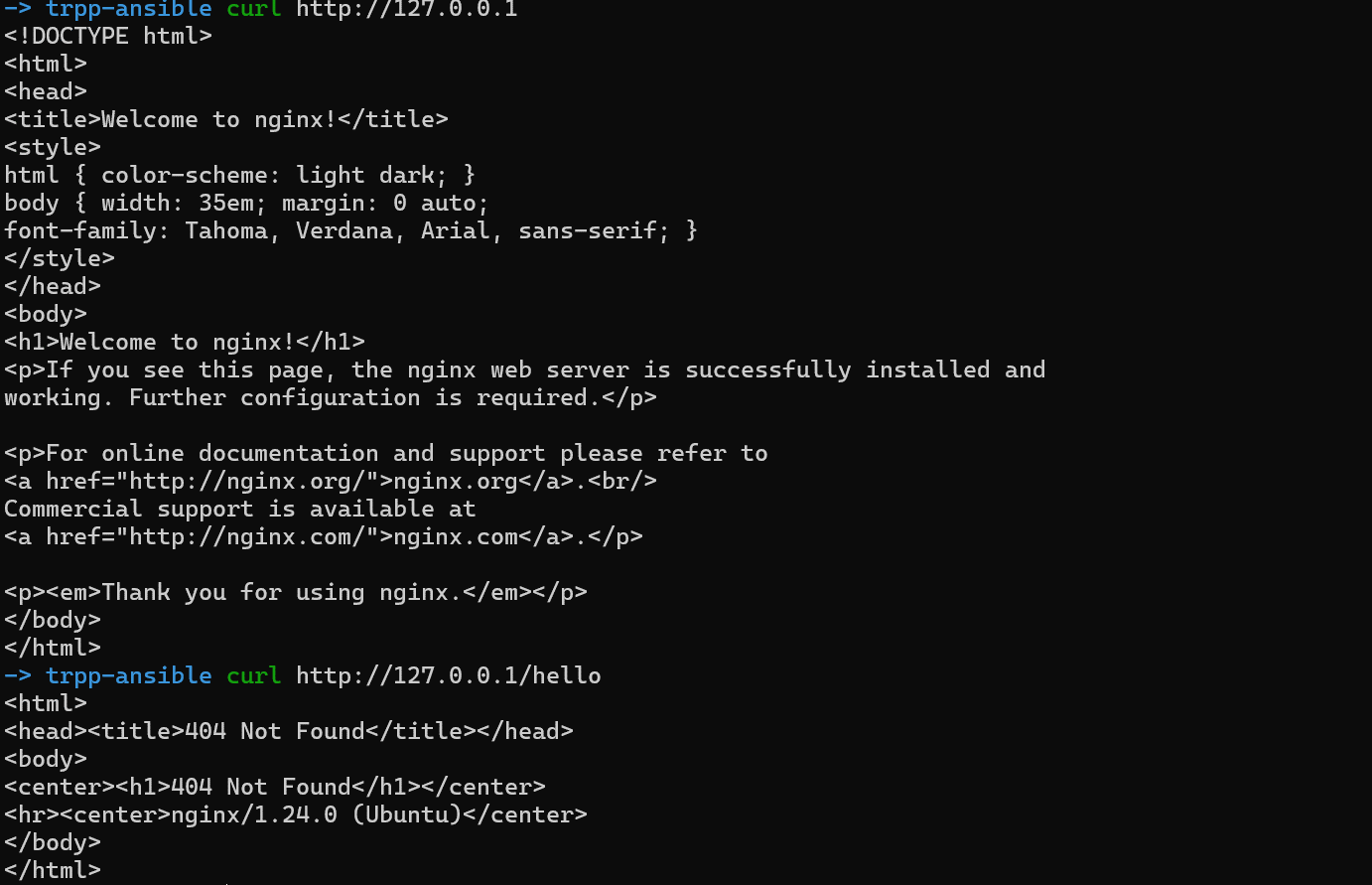


Рисунок 13 – Просмотр локального сайта конфигурации Nginx

**Роли Ansible**

Данный механизм позволяет систематизировать конфигурации путём выделения каждого механизма (задач, переменных, обработчиков и т. д.) в отдельные части.

Ansible имеет похожий на GitHub сервис, называемый Ansible Galaxy. Там находится множество ролей для Ansible, которыми можно воспользоваться. Для использования данного сервиса в Ansible встроена команда ansible-galaxy. Загрузим в систему роль для установки MySQL при помощи команды

*ansible-galaxy install geerlingguy.mysql*

Найти файлы установленной роли можно по следующему пути ~/.ansible/roles/geerlingguy.mysql/. Рассмотрим более подробно содержимое данной роли. При помощи команды tree . можно вывести дерево каталога для большой наглядности, рис 14.

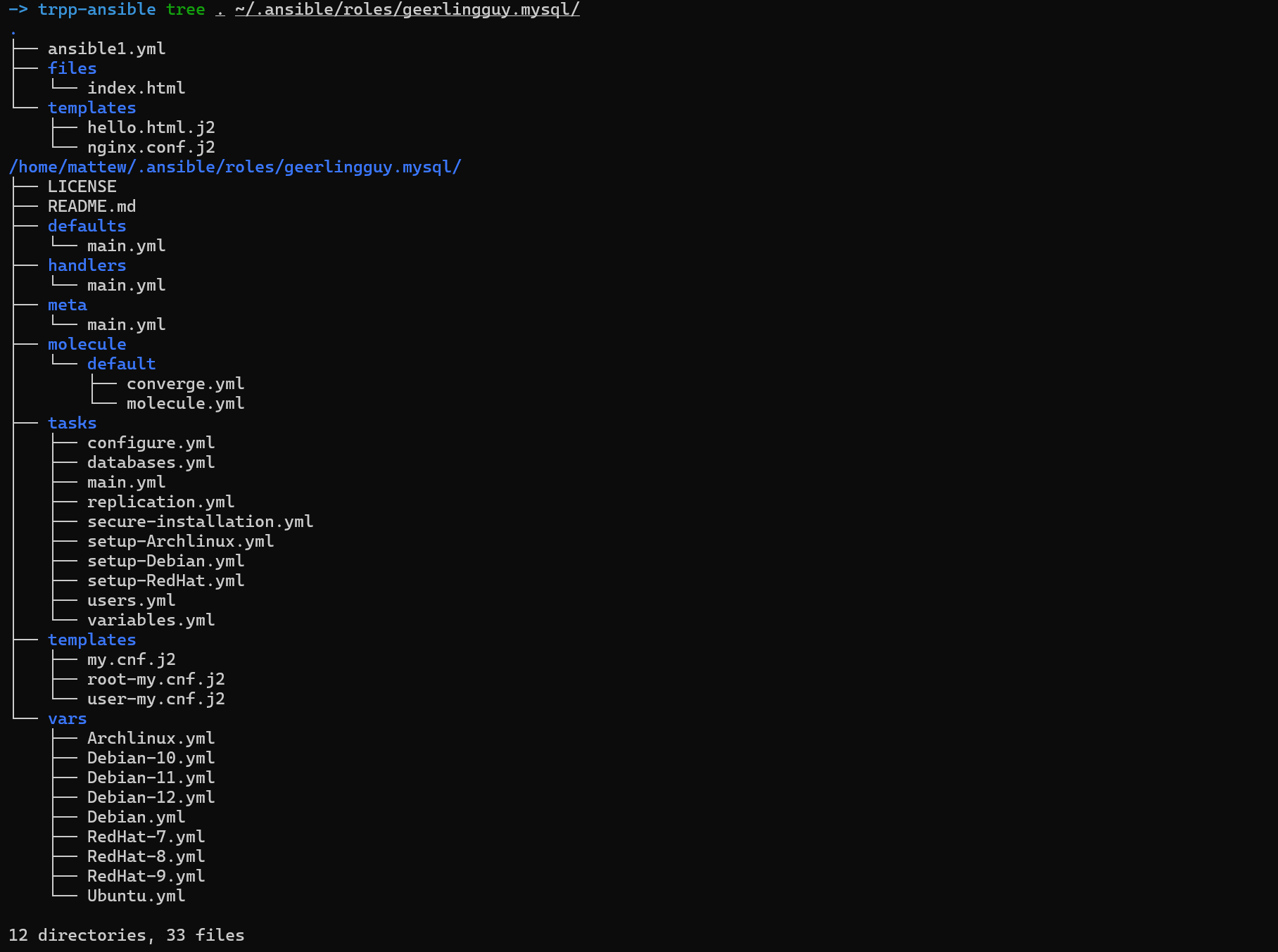


Рисунок 14 – Содержимое директории с ролью mysql

• Директория defaults содержит значения переменных по умолчанию.

• Директория handlers содержит описание обработчиков.

• Директория meta содержит информацию о роли, то есть создателе роли, её описание, используемая лицензия, зависимости и т. д. Данный файл используется в дальнейшем для Ansible Galaxy, который будет описан далее.

• Директория molecule содержит сценарии тестирования для роли Ansilbe.

• Директория tasks содержит непосредственно сценарии для конфигураций.

• Директория templates, как можно догадаться, содержит шаблонные параметризованные файлы.

• Директория vars содержит файлы описания различных переменных.

Это значит, что в директории tasks файл main.yml должен быть заполнен данными из секции tasks(рис 17). В директории defaults - из секции vars(рис 15). В директории handlers - из секции handlers(рис 16). Также файлы из директорий files и templates должны быть перемещены в директории files и templates в папке роли, рис 18.

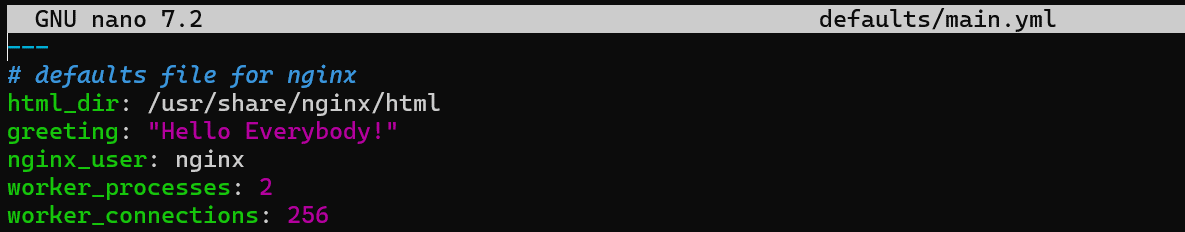


Рисунок 15 – Файл с переменными роли

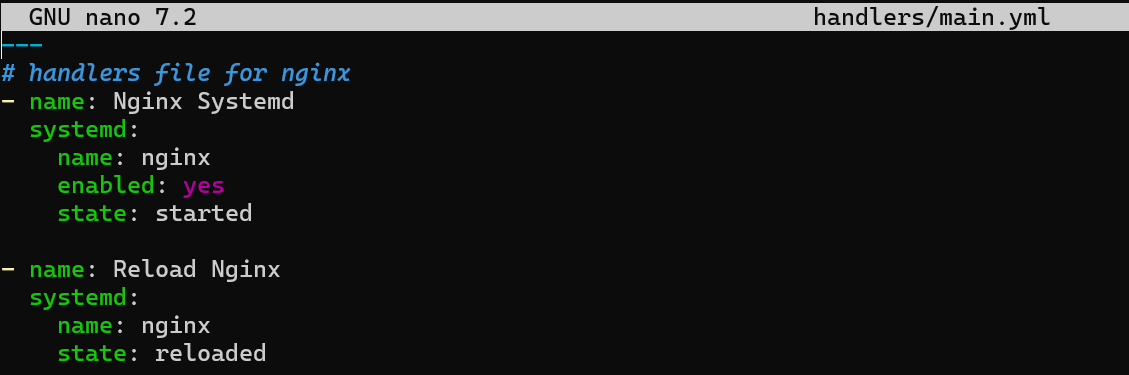


Рисунок 16 – Файл с handlers роли

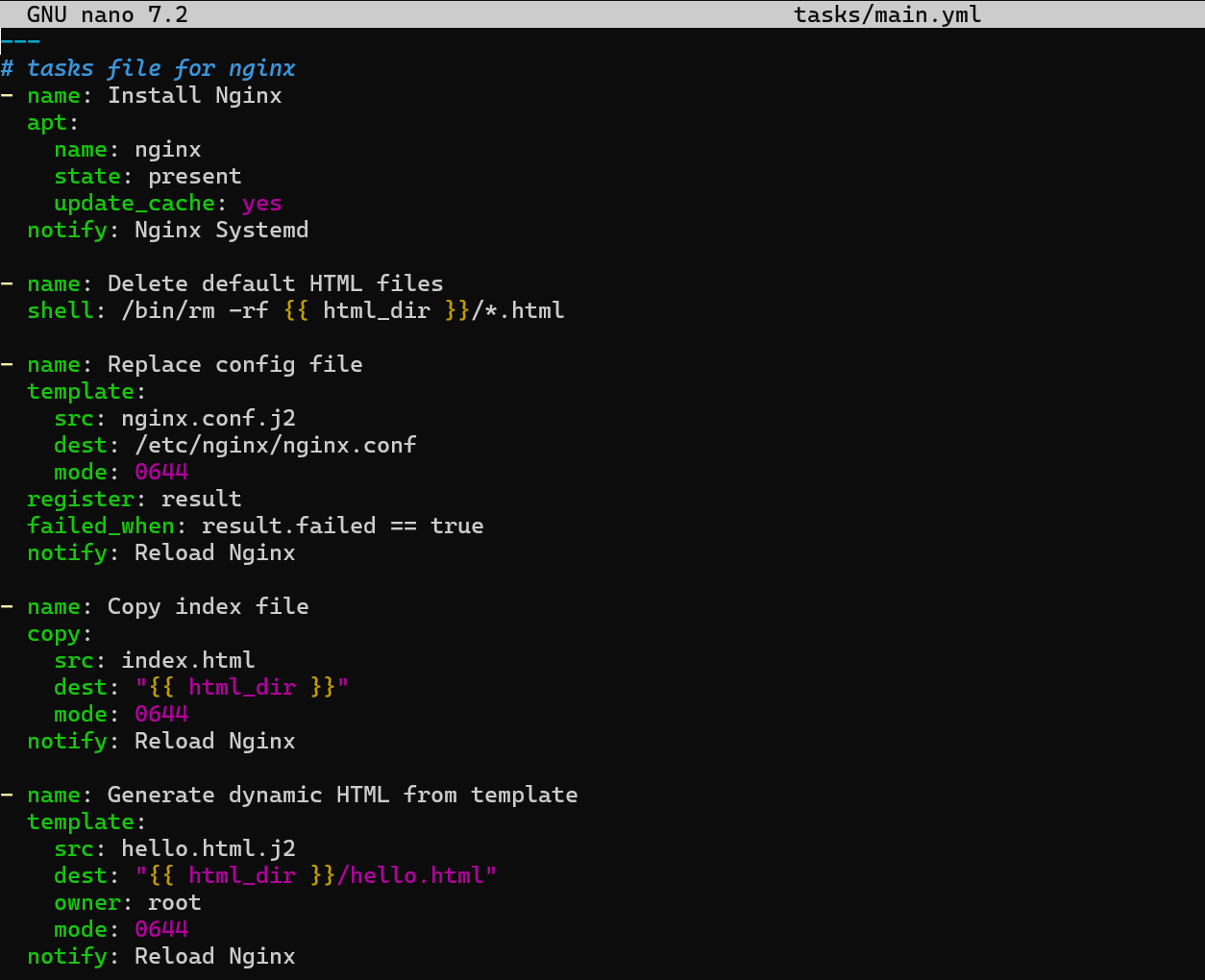


Рисунок 17 – Файл с tasks роли

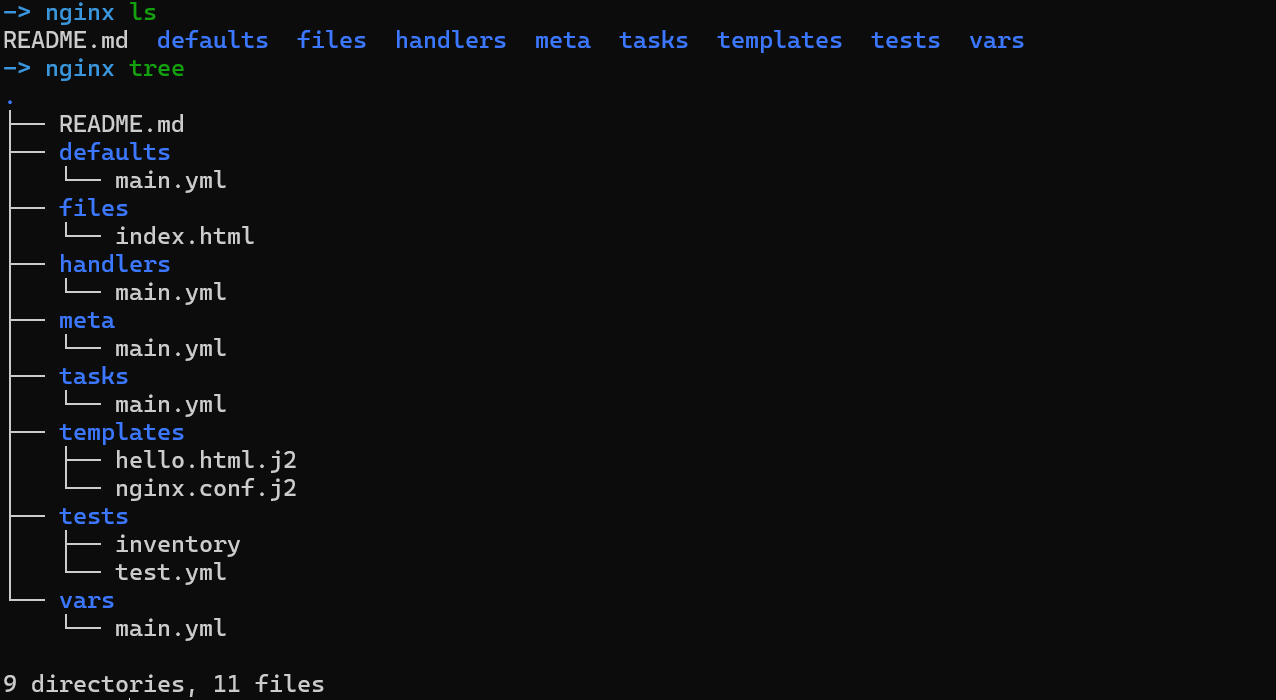


Рисунок 18 – Файловая система проекта

Результат выполнения роли отображен на рис 19.

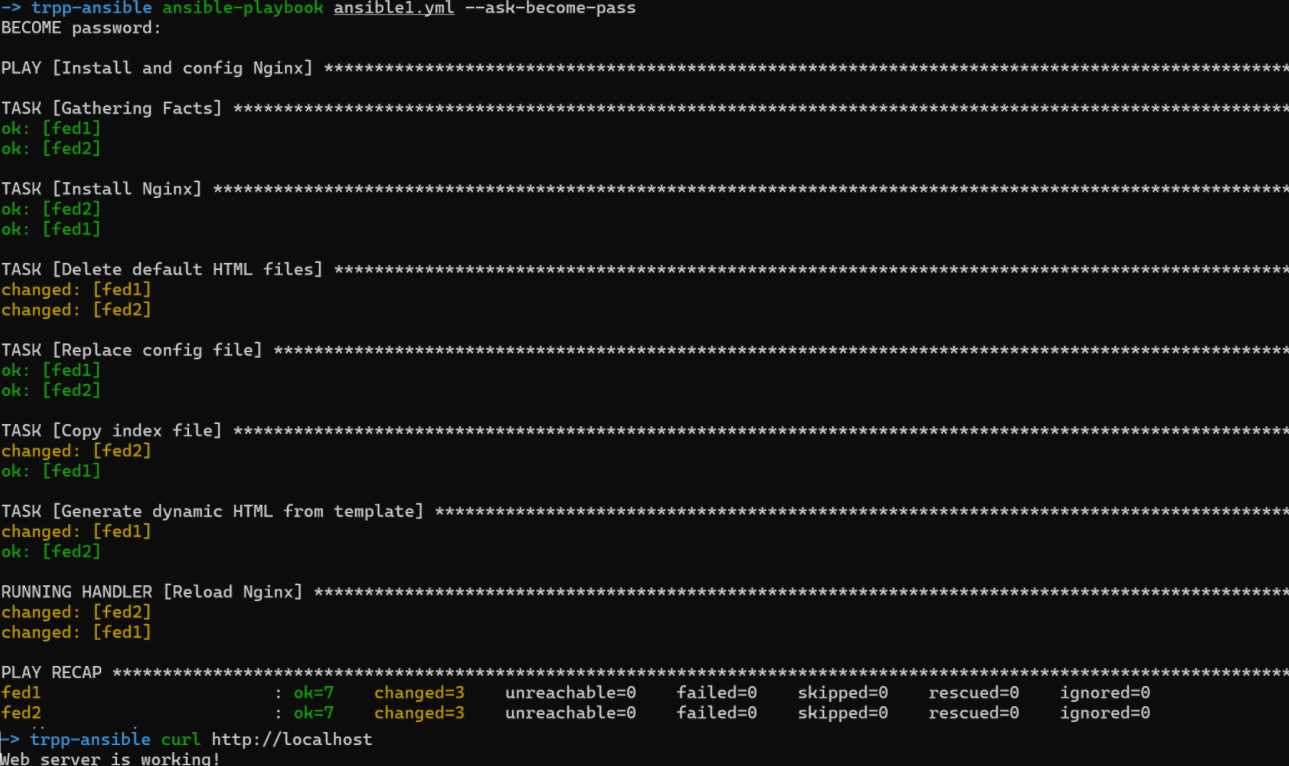


Рисунок 19 – Результат выполнения плейбука и роли

## Индивидуальное задание

Написать роль для запуска сервера nginx, написать playbook для применения роли, провести тестовый запуск playbook’a, в случае успешного прохождения теста, применить playbook к серверам.

Необходимо добавить переменную, содержащую ФИО, номер группы и номер варианта (Албахтин Илья Владиславович, ИНБО-12-23, 3 Вариант). Данная переменная должна выводиться в шаблонный файл nginx.

Установка пакета выполняется при помощи модуля APT, используемого для установки nginx в базовой роли.

Добавьте в playbook task по установке пакета согласно варианту, 3. wget

Изображение выглядит как текст, Шрифт, снимок экрана, дизайн

Контент, сгенерированный ИИ, может содержать ошибки.Изображение выглядит как текст, Шрифт, снимок экрана, дизайн

Контент, сгенерированный ИИ, может содержать ошибки.Изображение выглядит как текст, Шрифт, снимок экрана, дизайн

Контент, сгенерированный ИИ, может содержать ошибки.Изображение выглядит как текст, Шрифт, снимок экрана, дизайн

Контент, сгенерированный ИИ, может содержать ошибки.Измененный файл main.yml в директории defaults отображен на рис. 20.

Изображение выглядит как текст, Шрифт, снимок экрана, дизайн

Контент, сгенерированный ИИ, может содержать ошибки.Изображение выглядит как текст, Шрифт, снимок экрана, дизайн

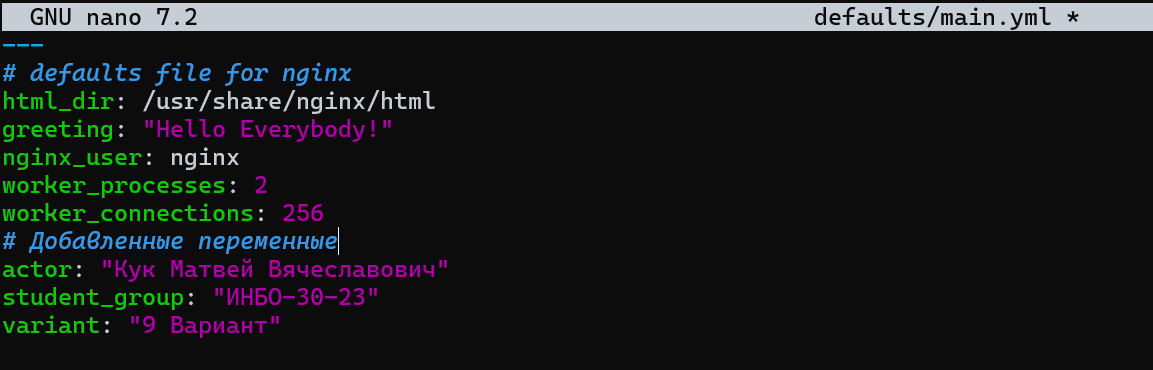
Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 20 – Новый файл defaults/main.yml

Измененный файл main.yml в директории tasks, рис 21.

Изображение выглядит как текст, Шрифт, снимок экрана, дизайн

Контент, сгенерированный ИИ, может содержать ошибки.Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 21 – Установка пакета wget

Для удобства и автоматизации использовалась утилита molecule, проводящая тесты роли и плейбука соответственно, рис 22.

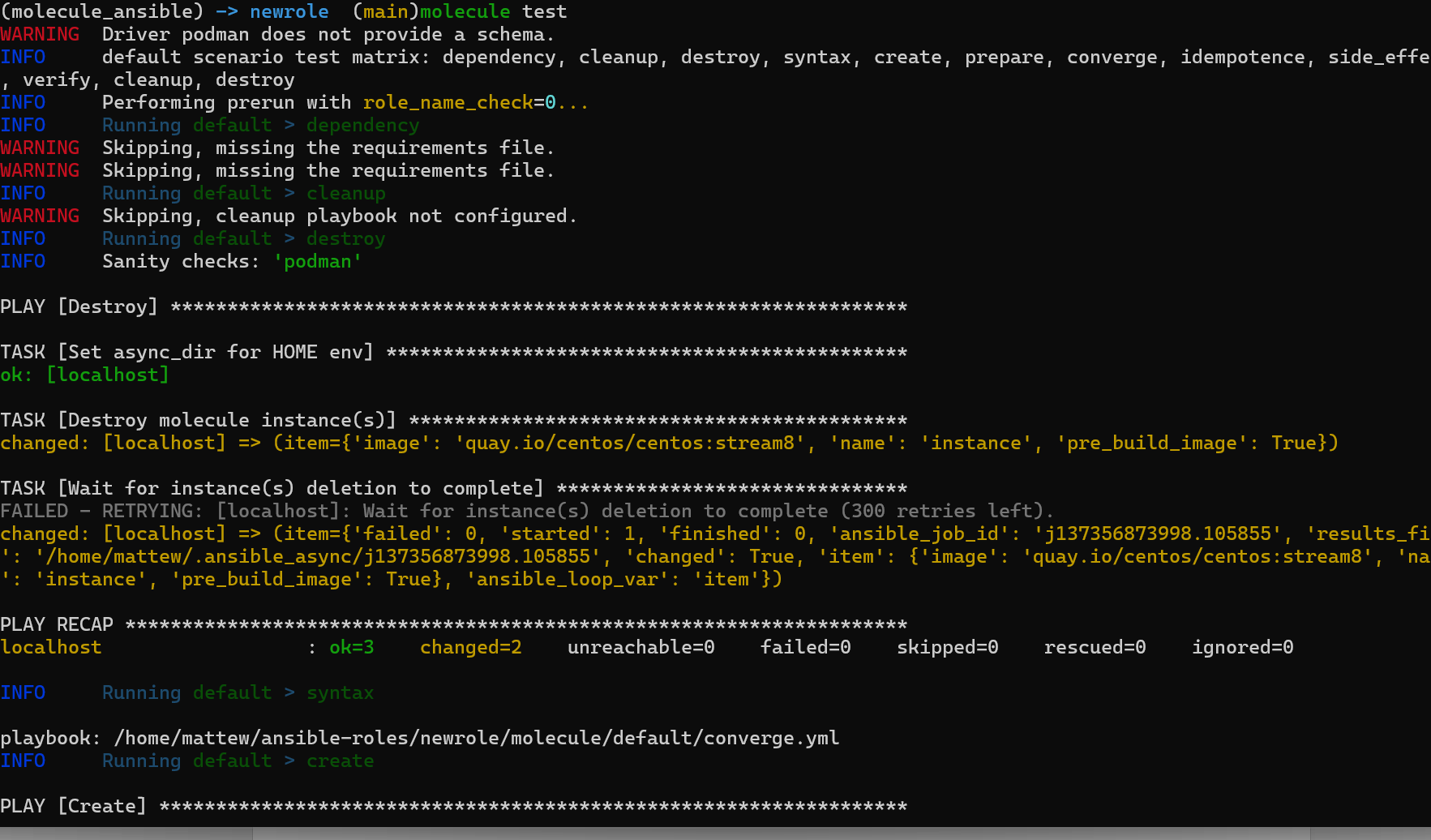


Рисунок 22 – Результат тестирования

Так как тестирование прошло успешно, можно запускать плейбук на управляемых узлах, рис 23.

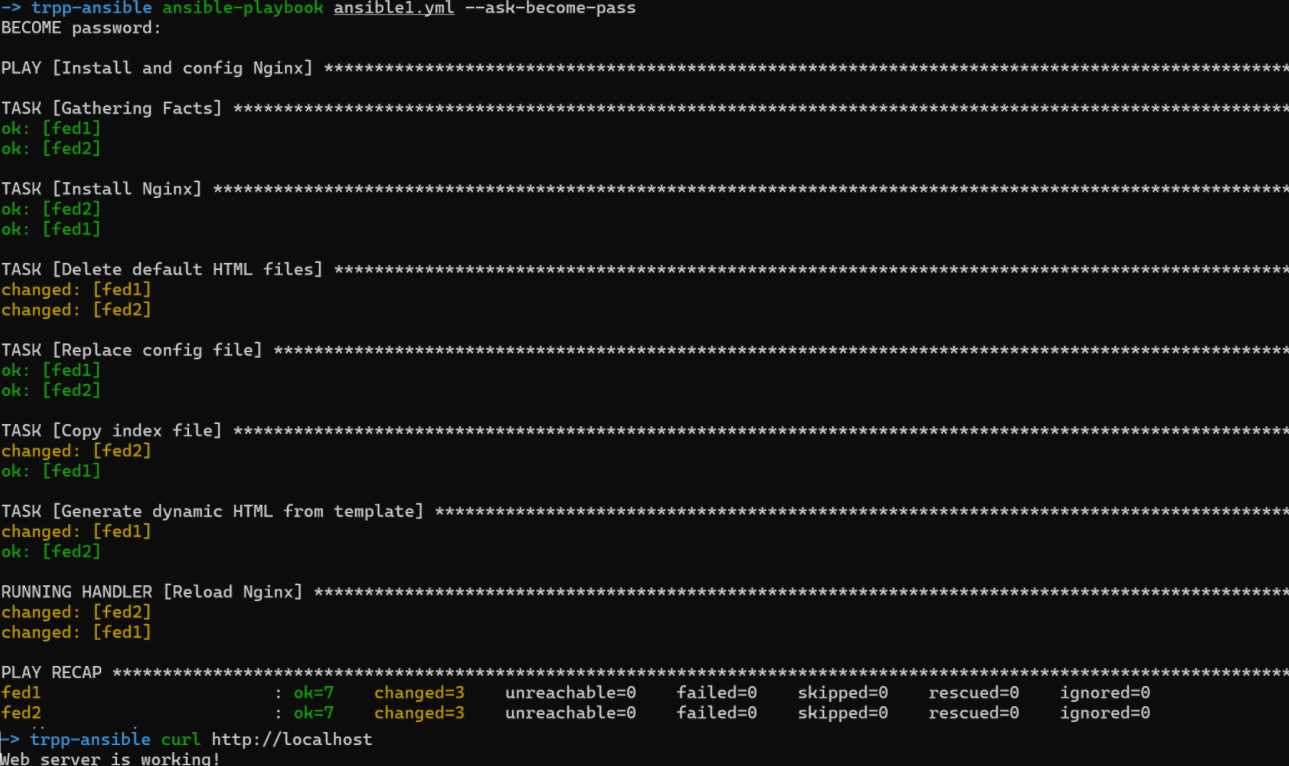


Рисунок 23 – Результат выполнения плейбука и роли

# Вывод

В ходе практической работы были получены навыки настройки вычислительной инфраструктуры с использованием системы управления конфигурациями Ansible. Созданы и настроены виртуальные машины, установлены необходимые компоненты, включая SSH и сам Ansible. Проведена базовая настройка inventory-файла, написан и успешно протестирован playbook для установки и конфигурации веб-сервера Nginx. Также реализована роль с использованием шаблонов и переменных, протестирована с помощью Molecule и применена к управляемым узлам. Работа позволила освоить как базовые, так и продвинутые возможности Ansible.