- **Name**: Iskander Nafikov
- **E-mail**: i.nafikov@innopolis.university
- **GitHub**: https://github.com/iskanred
- **DockerHub**: https://hub.docker.com/repository/docker/iskanred
- **Username**: `i.nafikov` / `iskanred`
- **Hostname**: `macbook-KN70WX2HPH` / `lenovo`

Task 1 - Prerequisites

> Prepare at least two different guest systems (VMs), e.g. an Ubuntu Server (aka "prod env") and a Fedora (aka "dev env"). Then choose a Software configuration management (SCM) tool to automate system preparations on those guests instances. Ansible is the default choice and the most popular tool for SCM. We recommend you to use Ansible.

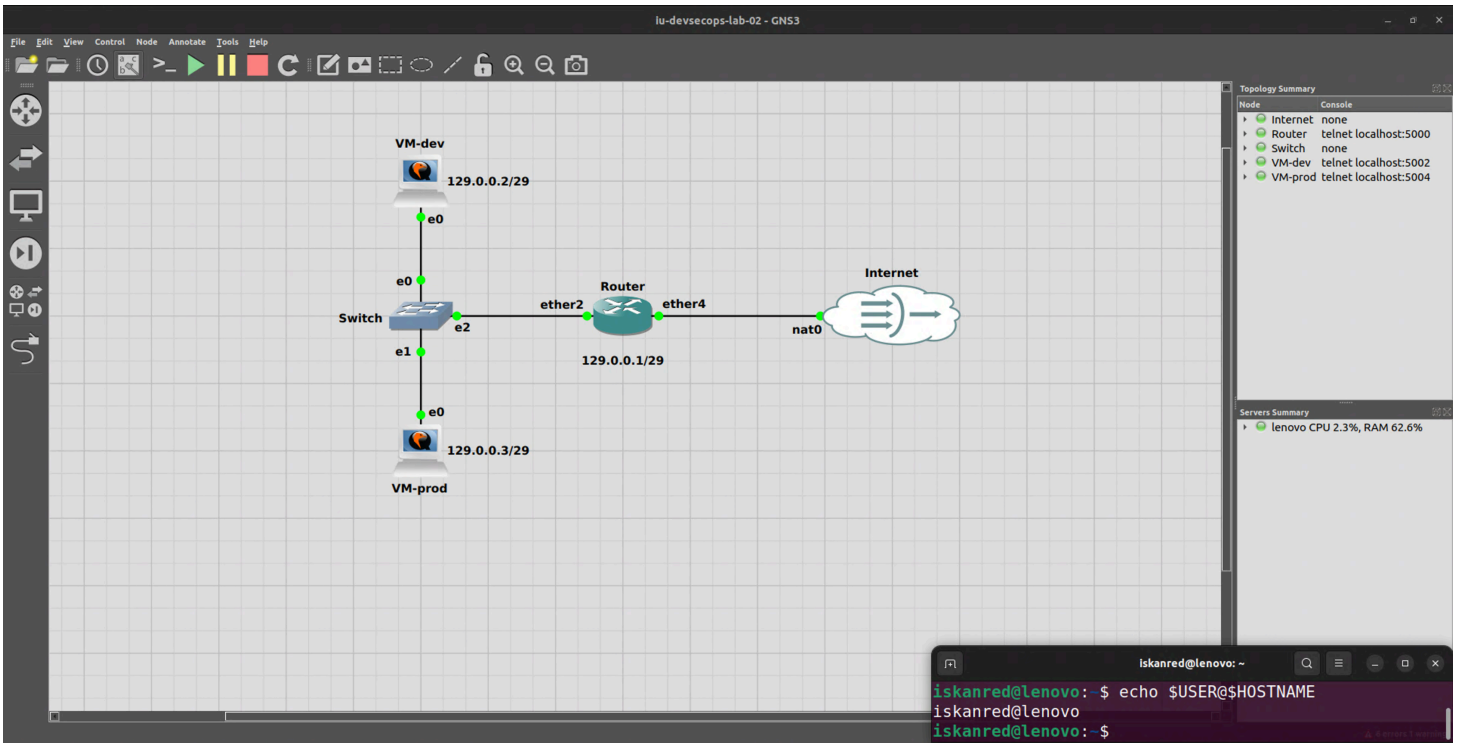
SCM choice

I used Ansible.

GNS3

To make the process of network setup **more explicit** I decided to use GNS3 because it gives a visual representation of a virtual network. I launched it on my `lenovo` laptop with Ubuntu Desktop.

- Below is the configuration of network topology that I created for this task:



Brief overview of nodes

- **`VM-dev`** (`129.0.0.2/29`) is a QEMU/KVM-virtualised Ubuntu Cloud 24.10 machine that acts as Ansible host machine in so called "dev" environment.
- **`VM-prod`** (`129.0.0.3/29`) is a QEMU/KVM-virtualised Ubuntu Cloud 24.10 machine that acts as Ansible host machine in so called "prod" environment.
- **`Router`** (`129.0.0.1/29`) is a QEMU/KVM-virtualised Mikrotik router that allows me to create LANs for my nodes easily.
- **`Internet`** is the node that allows all other nodes to interact with the internet via the **`Router`**. Actually it is the interface on my host machine (`lenovo`) with DHCP server running on it. To make access to internet possible the router uses NAT (src-nat) and DHCP (client) on the `ether4` interface. Also it allows me to connect from the outside (e.g. from `macbook-KN70WX2HPH`) to my VMs with configured static routes inside my LAN.

Finally, my connection to VMs looks like:

- Macbook (`macbook-KN70WX2HPH`) -> Home Router -> Lenovo (`lenovo`) -> Virtual Router (`Router`) -> VMs

---

Task 2 - SCM Theory

> Briefly answer for the following questions what is and for what:

1.

> **ansible** directory

- `ansible.cfg` : Configuration file for the specific Ansible project which defines settings for how Ansible operates, such as inventory locations, roles paths, privileges, and other environment-specific parameters.
- `inventory` : Specifies the hosts and groups of hosts that Ansible will manage, detailing how to connect to them.
- `roles` : Encapsulates tasks, variables, and handlers in a structured way, promoting reusability and modularity.
  - `tasks` : Directory within a role containing task definitions. Task is a minimal unit of managing in Ansible that is actually a set of commands.
  - `defaults/group_vars` : Directory for default variable definitions for the role / Directory for variables grouped by inventory groups. Specifies default values for variables to be used in roles, allowing customisation without modifying task files.
  - `handlers` : Lists tasks that are triggered by other tasks, typically used for actions that need to run only when notified, such as restarting a service.

    - `templates` : Contains template files that can be rendered with variables and replaced in target systems, typically for configuration files (e.g. docker-compose).
    - `vars` : Contains variable definitions specific to the role, which can override default values or provide specific configurations.

- `meta` : Contains information about the role (e.g., dependencies, author, description), which can help manage and document the role better.
- `playbooks` : Stores description of playbooks used to declare what tasks should be run on what hosts, organising the execution of roles and tasks in a structured manner.

2.

> Research, list and explain the most important parameter from `ansible.cfg` in your opinion

Precedence

Changes can be made and used in a configuration file which will be searched for in the following order:

- `ANSIBLE_CONFIG` (environment variable if set)
- `ansible.cfg` (in the current directory)
- `~/.ansible.cfg` (in the home directory)
- `/etc/ansible/ansible.cfg`

Important parameters

In my opinion, the most important and at the same time the most used parameters are:

- **`inventory`**

  > This parameter defines the location of the inventory files that Ansible will use to manage the hosts. The inventory serves as the **foundation of an Ansible automation**. It defines which machines are managed, their grouping, and connection details (e.g. inventory plugins), making it the first step in any Ansible operation.

  ```
  [defaults]
  inventory = /path/to/your/inventory
  ```

  This parameter can be set to a single inventory file, a folder containing multiple inventory files, or multiple comma-separated inventory sources. For example:
    - A single path: `/etc/ansible/hosts`
    - A directory: `/etc/ansible/inventory`
    - Multiple sources: `/path/to/inventory1,/path/to/inventory2`
- **`playbooks_dir`**

> This parameter defines the location of the directory that stores playbooks.

```
[defaults]
inventory = /path/to/your/playbooks
```

- **`remote_user`**

> Defines the default SSH user that Ansible will use to connect to remote hosts. **This is vital for authentication and access control**.

```
[defaults]
remote_user = ubuntu
```

- **`timeout`**

> Specifies how long (in seconds) Ansible will wait when attempting to connect to a host. This can be **useful in environments with varying network conditions**.

```
[defaults]
timeout = 10
```

- **`host_key_checking`**

> When set to `False`, it disables SSH host key checking. This can be **important in dynamic environments or CI/CD pipelines** but should be used cautiously from a security perspective.

```
[defaults]
host_key_checking = False
```

- **`any_unparsed_is_failed`**

> If 'true', it is a fatal error when any given inventory source cannot be successfully parsed by any available inventory plugin; otherwise, this situation only attracts a warning. It is especially useful for dynamic inventories!

```
[inventory]
any_unparsed_is_failed = True
```

3.

> Learn and explain Ansible variable precedence.

I found this information here on the Ansible official website.

Here is the order of precedence from **least to greatest** (the last listed variables override all other variables):

#	Name	Description
1	**Command Line Values**	Settings passed directly in the command line (e.g., `-u my_user`). Not considered variables like `-e` but important for configuration.
2	**Role Defaults**	Variables defined in the `{role_name}/defaults/main.yml` file within a role. Provides default values that can be overridden.
3	**Inventory File or Script Group Vars**	Variables defined for a group of hosts in inventory files or scripts, allowing for shared configurations among hosts in the same group.
4	**Inventory `group_vars/all`**	Variables defined in `group_vars/all`, which apply to all hosts in the inventory, serving as a common base configuration.
5	**Playbook `group_vars/all`**	Similar to inventory group vars, but specifically for group variables defined within a playbook context.
6	**Inventory `group_vars/*`**	Variables defined for specific groups located in the `group_vars/` directory, applying only to the respective groups.
7	**Playbook `group_vars/*`**	Variables defined for specific groups, scoped within a playbook (similar to inventory but allowing more tailored configurations).
8	**Inventory File or Script Host Vars**	Variables defined for individual hosts in the inventory file, enabling host-specific configurations.
9	**Inventory `host_vars/*`**	Variables specified in the `host_vars/` directory, which apply only to individual hosts, overriding group vars.
10	**Playbook `host_vars/*`**	Host variables defined within a playbook context, similar to individual inventory files but specific to plays.
11	**Host Facts / Cached `set_facts`**	Automatically collected facts about hosts or facts manually set during playbook runtime. Read-only and gathered via the `setup` module.
12	**Play Vars**	Variables defined using the `vars:` section within a play, applying specifically to the tasks within that play.
13	**Play `vars_prompt`**	Variables that are prompted from users at runtime, allowing dynamic input for playbooks.
14	**Play `vars_files`**	Variables sourced from external YAML files included in a play, making it easy to manage configurations outside of the playbookcontext.
15	**Role Vars**	Variables defined in the `vars/main.yml` file within a role. These vars have a higher precedence than role defaults and can override them.

#	Name	Description
16	**Block Vars**	Variables defined within a block in a playbook that apply only to the tasks within that block. Useful for scoping variables more tightly.
17	**Task Vars**	Variables defined at the task level, applying only to that specific task. This allows for very localized variable usage.
18	`include_vars`	Variables included from YAML files using the `include_vars` module. These vars can be scoped to specific tasks or whole plays based on where they are included.
19	**Set Facts / Registered Vars**	Variables created during playbook execution through the `set_fact` module or variables registered from task output. These take precedence over most definitions and are dynamic.
20	**Role (and `include_role`) Params**	Parameters passed to roles or roles included using `include_role`. These values apply specifically to the execution of that role.
21	**Include Params**	Parameters passed to `include` or `import` statements that apply to the tasks included or imported.
22	**Extra Vars (`-e`)**	Variables passed via the command line using the `-e` option. These variables take the highest precedence and can override all other variable definitions.

Task 3 - Play with SCM

1.

> Play with SCM tool to automate system preparations on those. You might create and provide your own ideas or follow to some suggestions.

Preparations

Configuring SSH

To make SSH connection establishment used by Ansible more secure I decided to connect with private SSH key instead of password.

- Firstly, I created two SSH key pairs for **`VM-prod`** and **`VM-dev`** hosts respectively using ED25519 algorithm

```
sudo ssh-keygen -t ed25519 -C "vm-dev"
sudo ssh-keygen -t ed25519 -C "vm-prod"
```

```
Terminal  Local
i.nafikov@macbook-KN70WX2HPH lab-02 % ls ~.ssh/ | grep id_ed25519_vm
zsh: no such user or named directory: .ssh
i.nafikov@macbook-KN70WX2HPH lab-02 % ls ~/.ssh/ | grep id_ed25519_vm
id_ed25519_vm_dev
id_ed25519_vm_dev.pub
id_ed25519_vm_prod
id_ed25519_vm_prod.pub
i.nafikov@macbook-KN70WX2HPH lab-02 %
```

- Then, I copied public keys to the corresponding hosts

```
ssh-copy-id -i id_ed25519_vm_dev.pub ubuntu@129.0.0.2
ssh-copy-id -i id_ed25519_vm_prod.pub ubuntu@129.0.0.3
```

```
Terminal  Local
ubuntu@ubuntu-cloud:~$ ip a | grep 129.0.0.2
    inet 129.0.0.2/29 brd 129.0.0.7 scope global ens3
ubuntu@ubuntu-cloud:~$ cat .ssh/authorized_keys
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIN+Bm0nO2Lo7FohGTkvTULkEBfxfWxq7ZF0wgUax15bh vm-dev
ubuntu@ubuntu-cloud:~$
```

```
Terminal  Local
ubuntu@ubuntu-cloud:~$ ip a | grep 129.0.0.3
    inet 129.0.0.3/24 brd 129.0.0.255 scope global ens3
ubuntu@ubuntu-cloud:~$ cat .ssh/authorized_keys
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIEUgV6hC2hgqi4mvVtCBGtHzDNY4At2j3jiqbPAFDSx0 vm-prod
ubuntu@ubuntu-cloud:~$
```

- Finally, I could connect from my host machine (`macbook-KN70WX2HPH`) to the VMs without password

```
Terminal  Local
i.nafikov@macbook-KN70WX2HPH lab-02 % ssh -i ~/.ssh/id_ed25519_vm_dev ubuntu@129.0.0.2
Welcome to Ubuntu 24.10 (GNU/Linux 6.11.0-8-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Tue Feb  4 18:25:45 UTC 2025

  System load:  0.0               Processes:             96
  Usage of /:   71.6% of 2.25GB   Users logged in:       0
  Memory usage: 17%               IPv4 address for ens3: 129.0.0.2
  Swap usage:   0%


0 updates can be applied immediately.


The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Tue Feb  4 18:14:37 2025 from 192.168.1.1
ubuntu@ubuntu-cloud:~$
```

```
Terminal  Local
i.nafikov@macbook-KN70WX2HPH lab-02 % ssh -i ~/.ssh/id_ed25519_vm_prod ubuntu@129.0.0.3
Welcome to Ubuntu 24.10 (GNU/Linux 6.11.0-8-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Tue Feb  4 18:27:31 UTC 2025

  System load:  0.0               Processes:             97
  Usage of /:   71.6% of 2.25GB   Users logged in:       0
  Memory usage: 18%               IPv4 address for ens3: 129.0.0.3
  Swap usage:   0%


0 updates can be applied immediately.


The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Tue Feb  4 18:14:42 2025 from 192.168.1.1
ubuntu@ubuntu-cloud:~$
```

Ansible setup

- To begin, I created a working directory for Ansible: `iu-devsecops-course/lab-02/ansible`

```
Terminal  Local
i.nafikov@macbook-KN70WX2HPH lab-02 % ls | grep ansible
ansible
i.nafikov@macbook-KN70WX2HPH lab-02 %
```

- Then I had to install `ansible` package through the `pip`. To do so I created virtual Python environment `ansible-env` inside the `ansible` directory

```
cd ansible
python3 -m venv ansible-env
source ansible-env/bin/activate
```

```
Terminal  Local
(ansible-env) i.nafikov@macbook-KN70WX2HPH ansible %
```

- I installed `ansible` package

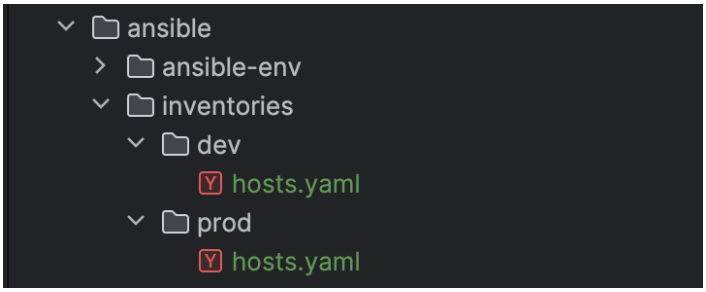
```
python3 -m pip install --upgrade pip
python3 -m pip install ansible
```

```
Terminal  Local
(ansible-env) i.nafikov@macbook-KN70WX2HPH ansible % which ansible
/Users/i.nafikov/Study/iu-devsecops-course/lab-02/ansible/ansible-env/bin/ansible
(ansible-env) i.nafikov@macbook-KN70WX2HPH ansible % python3 -m pip list | grep ansible
ansible      11.2.0
ansible-core 2.18.2
(ansible-env) i.nafikov@macbook-KN70WX2HPH ansible %
```

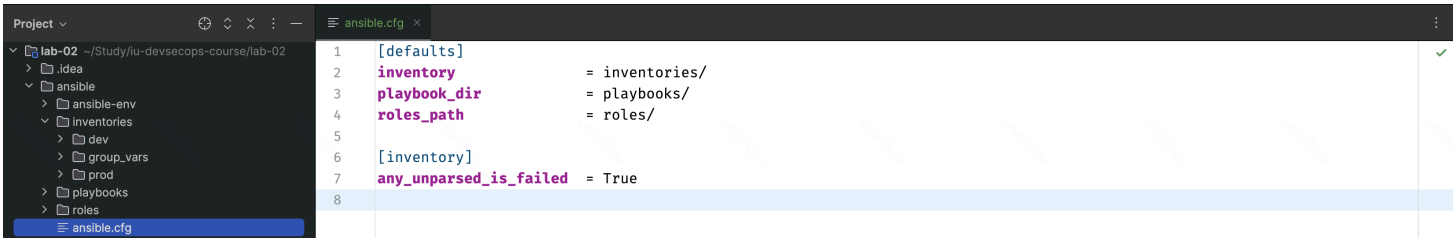
Ansible

Inventories configuration

- Firstly, I needed to setup inventories
- For this purpose I created the following file structure for `dev` and `prod` environments

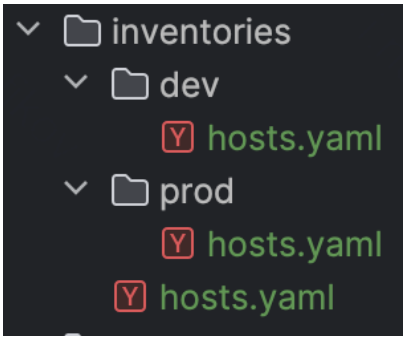


- And I had to say Ansible in `ansible.cfg` where to find my inventories' configuration together with playbooks' and roles'. As you can see also I added a useful option `any_unparsed_is_failed`.



- Afterward, I setup my `hosts.yaml` files. I made up 3 groups: `webserver`, `prod`, `dev`. The groups `prod` and `dev` are children of `server`. Also, I added location of SSH private keys as a **host variable**

and SSH username as a **group variable** for `server`.




```
        all
         |
       server
        /\
       /  \
    prod  dev
```

dev/hosts.yml

```yaml
server:
  children:
    dev:
      hosts:
        129.0.0.2:
          ansible_ssh_private_key_file: ~/.ssh/id_ed25519_vm_dev
```

prod/hosts.yml

```yaml
server:
  children:
    prod:
      hosts:
        129.0.0.3:
          ansible_ssh_private_key_file: ~/.ssh/id_ed25519_vm_prod
```

- From this moment I could connect to my hosts via Ansible

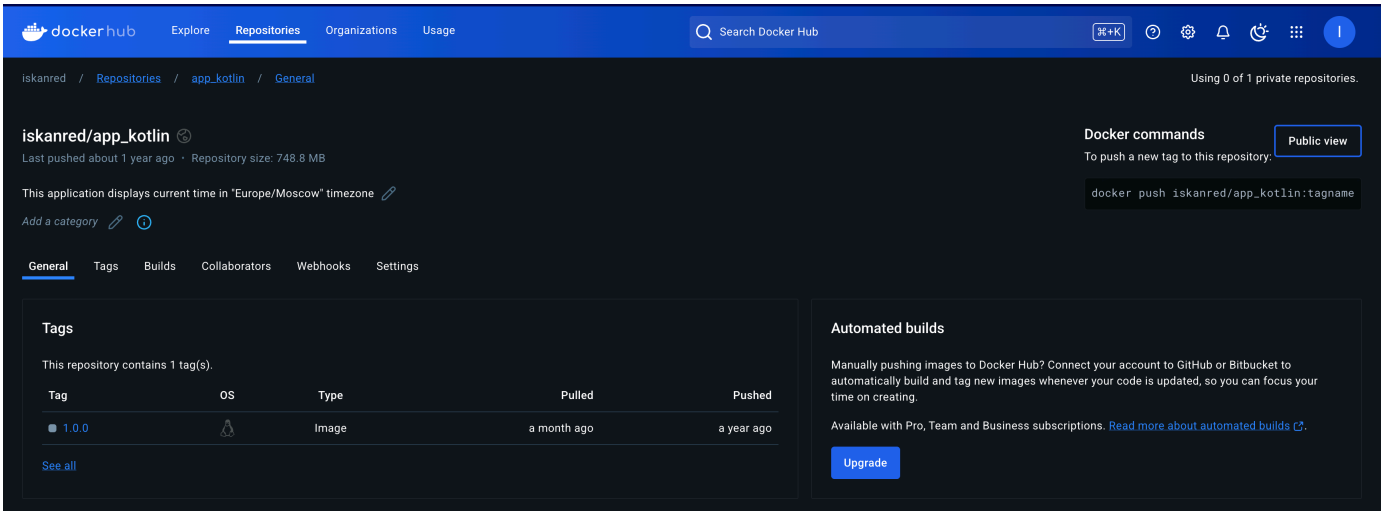
```
ansible all -m ping
```

```
Terminal  Local
(ansible-env) i.nafikov@macbook-KN70WX2HPH ansible % ansible all -m ping
[WARNING]: Platform linux on host 129.0.0.2 is using the discovered Python interpreter at /usr/bin/python3.12, but future installation of another Python
interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-core/2.18/reference_appendices/interpreter_discovery.html for more
information.
129.0.0.2 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3.12"
    },
    "changed": false,
    "ping": "pong"
}
[WARNING]: Platform linux on host 129.0.0.3 is using the discovered Python interpreter at /usr/bin/python3.12, but future installation of another Python
interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-core/2.18/reference_appendices/interpreter_discovery.html for more
information.
129.0.0.3 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3.12"
    },
    "changed": false,
    "ping": "pong"
}
(ansible-env) i.nafikov@macbook-KN70WX2HPH ansible %
```

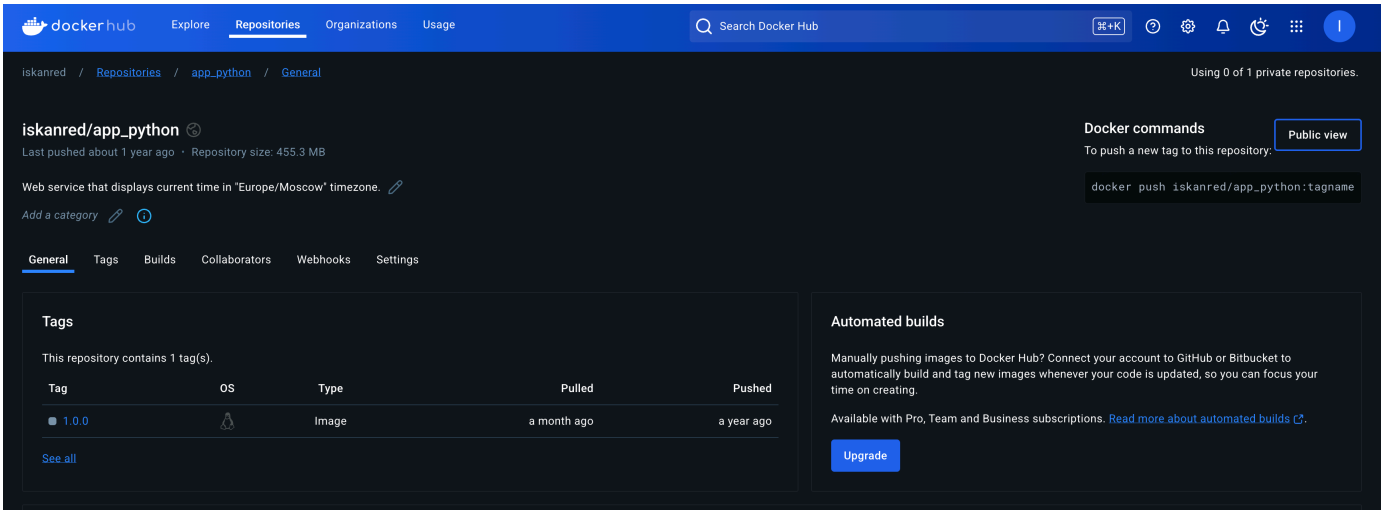
Project choice & Requirements

I had to select what will I deploy to my VMs using Ansible. There were a lot of options and I decided to create a project that would combine both "light" and "advanced" automations in a form of single deployment service. Let's see what I've come to:

- I had already had two identical simple web applications that display current `Europe/Moscow` time and they had been already pushed in a form of Docker images to DockerHub.
  - One of them is written in **Kotlin/JVM**: https://hub.docker.com/repository/docker/iskanred/app_kotlin/general



  - Another one is written in **Python**: https://hub.docker.com/repository/docker/iskanred/app_python



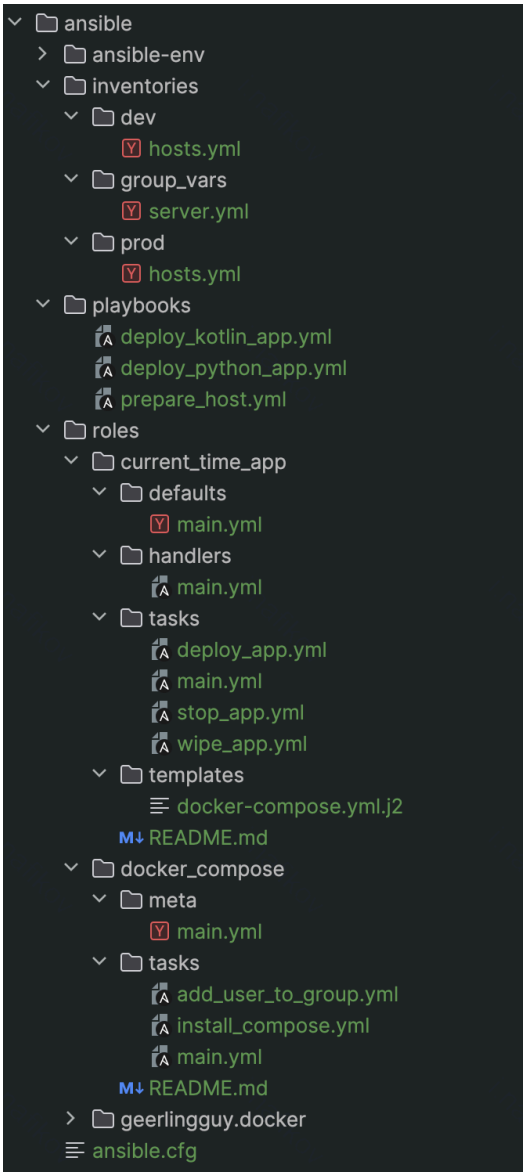
- I decided that I would deploy these applications to my VMs!
- Let's imagine that by some reason this, let's call it "Current Time", service must be deployed to the different environments in different forms:
  - Let `VM_dev` receive Python application
  - Let `VM_prod` receive Kotlin application
- These applications use bind mount volume to keep the number of website visits, port mapping, configure restart policy and etc. Therefore, to simplify deployment process I decided to use Docker Compose for containers deployment.
  - To work with `docker-compose` command I used `ansible.builtin.command` module.
- To achieve separation of the applications I made a common **Jinja2 template** for `docker-compose.yaml` for both applications inside the `current_time_app` role. Depending on variables used in playbooks we could deploy either Python or Kotlin application to the hosts we needed.
  - Therefore, I needed to use `ansible.builtin.template` module.
  - In addition, to manage files and directories I used `ansible.builtin.file` and `ansible.builtin.stat` modules.

- Therefore, besides everything mentioned I had to install: `docker` and `docker-compose` to both of my VMs.
  - For Docker I used a ready **[`geerlingguy.docker`]**(https://github.com/geerlingguy/ansible-role-docker) role.
  - For Docker Compose I created my own `docker_compose` that installs Docker Compose using `apt`
    - For `apt` I used `ansible.builtin.apt` module.
    - Also, I needed to add my user to `docker` Linux group to be able to run containers without a privileged access. To achieve this I used ansible.builtin.user module.
- What is more, I wanted to have some possibilities to control the application's lifecycle using variables:
  - A possibility to stop the currently running application before a new deployment.
  - Since my applications are stateful (they keep number of website visits in a file) I wanted to have a possibility to wipe (clear) an application's files before a new deployment.

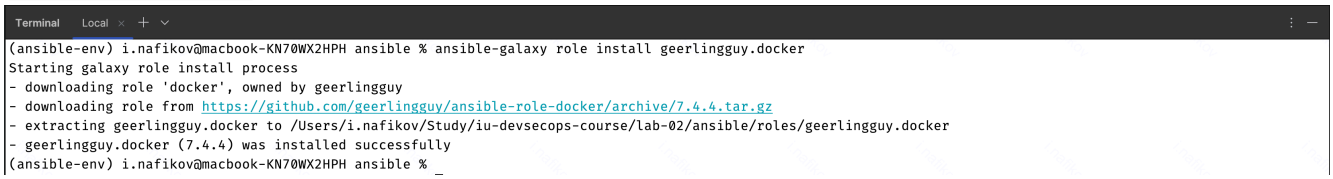
Implementation overview

Let's check what have been done.

- Firstly, I want to present my final file structure



- We can see that in total I have
  - **3 playbooks**:
    1. `deploy_kotlin_app` which deploys Kotlin app on `VM-prod`.
    2. `deploy_python_app` which deploys Python app on `VM-dev`.
    3. `prepare_host` which installs Docker and Docker Compose on `all` the VMs. This playbooks has privileged access for executing commands (`become: true`) because it interacts with **apt** (updating, installing packages) and modifies Linux user groups.
  - **3 roles**:
    1. `current_time_app` that install Docker Compose on a server. This role consists of 3 "separated" tasks:
       1. `stop_app` which can **optionally** stop an application before deploying a newer version. Optionality is configured using `web_app_full_stop` boolean variable in a playbook. It can only be done if `docker-compose.yml` exists, so there is a check of existence.
       2. `wipe_app` which can **optionally** clear an application's files before deploying a newer version. Optionality is configured using `web_app_full_wipe` boolean variable a playbook. It can only be done if `docker-compose.yml` exists, so there is a check of existence.
       3. `deploy_app` which creates application's directory, creates `docker-compose.yml` from the **Jinja2** template, and notifies `docker_compose_restart_handler` to run this `docker-compose.yml` configuration.
    2. `docker_compose` that deploys a "Current Time" application to a server. This role consists of 2 "separated" tasks:
       1. `install_compose` which installs Docker Compose Python module to deploy any Compose file.
       2. `add_user_to_group` which adds my user to `docker` Linux group to be able to run containers without a privileged access.
          This role also depends on the `geerlingguy.docker` role because Docker Compose has no sense without Docker. This dependency was specified inside the `roles/docker_compose/meta.yml`
    3. `geerlingguy.docker` that installs Docker on a server. This role was installed using `ansible-galaxy`.



- ❗ By "separated" tasks I mean tasks that are separated in its own YAML files while they can include another tasks inside these files.
- I put a detailed description of each role in `README.md` files in the corresponding directories:
  - `docker_compose`: lab-02/ansible/roles/docker_compose/README.md
  - `current_time_app`: lab-02/ansible/roles/current_time_app/README.md

Run

Finally, we can run our playbooks.

- Firstly, I run `prepare_host` playbook

prepare_host.yml

```yaml
- name: Prepare Host
  hosts: all
  become: true
  roles:
    - docker_compose
    - geerlingguy.docker

```

```
ansible-playbook playbooks/prepare_host.yml
```

```
Terminal  Local
(ansible-env) i.nafikov@macbook-KN70WX2HPH ansible % ansible-playbook playbooks/prepare_host.yml

PLAY [Prepare Host] ************************************************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************************************
[WARNING]: Platform linux on host 129.0.0.3 is using the discovered Python interpreter at /usr/bin/python3.12, but future installation of another Python
interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-core/2.18/reference_appendices/interpreter_discovery.html for more
information.
ok: [129.0.0.3]
[WARNING]: Platform linux on host 129.0.0.2 is using the discovered Python interpreter at /usr/bin/python3.12, but future installation of another Python
interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-core/2.18/reference_appendices/interpreter_discovery.html for more
information.
ok: [129.0.0.2]

TASK [geerlingguy.docker : Load OS-specific vars.] *****************************************************************************************************************
ok: [129.0.0.2]
ok: [129.0.0.3]

TASK [geerlingguy.docker : include_tasks] **************************************************************************************************************************
skipping: [129.0.0.2]
skipping: [129.0.0.3]

TASK [geerlingguy.docker : include_tasks] **************************************************************************************************************************
included: /Users/i.nafikov/Study/iu-devsecops-course/lab-02/ansible/roles/geerlingguy.docker/tasks/setup-Debian.yml for 129.0.0.2, 129.0.0.3

TASK [geerlingguy.docker : Ensure apt key is not present in trusted.gpg.d] *****************************************************************************************
ok: [129.0.0.3]
ok: [129.0.0.2]

TASK [geerlingguy.docker : Ensure old apt source list is not present in /etc/apt/sources.list.d] ******************************************************************
ok: [129.0.0.2]
ok: [129.0.0.3]

TASK [geerlingguy.docker : Ensure the repo referencing the previous trusted.gpg.d key is not present] *************************************************************
ok: [129.0.0.2]
ok: [129.0.0.3]

TASK [geerlingguy.docker : Ensure old versions of Docker are not installed.] ***************************************************************************************
```

- The whole output of this command was:

```
PLAY [Prepare Host]
************************************************************************************
************************************************************

TASK [Gathering Facts]
************************************************************************************
************************************************************
[WARNING]: Platform linux on host 129.0.0.3 is using the discovered Python
interpreter at /usr/bin/python3.12, but future installation of another Python
interpreter could change the meaning of that path. See
```

```
https://docs.ansible.com/ansible-
core/2.18/reference_appendices/interpreter_discovery.html for more
information.
ok: [129.0.0.3]
[WARNING]: Platform linux on host 129.0.0.2 is using the discovered Python
interpreter at /usr/bin/python3.12, but future installation of another Python
interpreter could change the meaning of that path. See
https://docs.ansible.com/ansible-
core/2.18/reference_appendices/interpreter_discovery.html for more
information.
ok: [129.0.0.2]

TASK [geerlingguy.docker : Load OS-specific vars.]
*********************************************************************************
*****************************
ok: [129.0.0.2]
ok: [129.0.0.3]

TASK [geerlingguy.docker : include_tasks]
*********************************************************************************
*************************************
skipping: [129.0.0.2]
skipping: [129.0.0.3]

TASK [geerlingguy.docker : include_tasks]
*********************************************************************************
*************************************
included: /Users/i.nafikov/Study/iu-devsecops-course/lab-
02/ansible/roles/geerlingguy.docker/tasks/setup-Debian.yml for 129.0.0.2,
129.0.0.3

TASK [geerlingguy.docker : Ensure apt key is not present in trusted.gpg.d]
*********************************************************************************
*****
ok: [129.0.0.3]
ok: [129.0.0.2]

TASK [geerlingguy.docker : Ensure old apt source list is not present in
/etc/apt/sources.list.d]
*********************************************************************
ok: [129.0.0.2]
ok: [129.0.0.3]

TASK [geerlingguy.docker : Ensure the repo referencing the previous trusted.gpg.d
```

```
key is not present] ************************************************************
ok: [129.0.0.2]
ok: [129.0.0.3]

TASK [geerlingguy.docker : Ensure old versions of Docker are not installed.]
********************************************************************************
***
ok: [129.0.0.3]
ok: [129.0.0.2]

TASK [geerlingguy.docker : Ensure dependencies are installed.]
********************************************************************************
******************
ok: [129.0.0.3]
changed: [129.0.0.2]

TASK [geerlingguy.docker : Ensure directory exists for /etc/apt/keyrings]
********************************************************************************
******
ok: [129.0.0.3]
ok: [129.0.0.2]

TASK [geerlingguy.docker : Add Docker apt key.]
********************************************************************************
*********************************
ok: [129.0.0.3]
changed: [129.0.0.2]

TASK [geerlingguy.docker : Ensure curl is present (on older systems without SNI).]
********************************************************************************
skipping: [129.0.0.2]
skipping: [129.0.0.3]

TASK [geerlingguy.docker : Add Docker apt key (alternative for older systems
without SNI).]
********************************************************************************
skipping: [129.0.0.2]
skipping: [129.0.0.3]

TASK [geerlingguy.docker : Add Docker repository.]
********************************************************************************
*****************************
ok: [129.0.0.3]
changed: [129.0.0.2]
```

```
TASK [geerlingguy.docker : Install Docker packages.]
*****************************************************************************
***************************
skipping: [129.0.0.2]
skipping: [129.0.0.3]

TASK [geerlingguy.docker : Install Docker packages (with downgrade option).]
*****************************************************************************
***
ok: [129.0.0.3]
changed: [129.0.0.2]

TASK [geerlingguy.docker : Install docker-compose plugin.]
*****************************************************************************
**********************
skipping: [129.0.0.2]
skipping: [129.0.0.3]

TASK [geerlingguy.docker : Install docker-compose-plugin (with downgrade option).]
*****************************************************************************
ok: [129.0.0.2]
ok: [129.0.0.3]

TASK [geerlingguy.docker : Ensure /etc/docker/ directory exists.]
*****************************************************************************
***************
skipping: [129.0.0.2]
skipping: [129.0.0.3]

TASK [geerlingguy.docker : Configure Docker daemon options.]
*****************************************************************************
*******************
skipping: [129.0.0.2]
skipping: [129.0.0.3]

TASK [geerlingguy.docker : Ensure Docker is started and enabled at boot.]
*****************************************************************************
******
ok: [129.0.0.3]
ok: [129.0.0.2]

TASK [geerlingguy.docker : Ensure handlers are notified now to avoid firewall
conflicts.]
```

```
*********************************************************************

TASK [geerlingguy.docker : Ensure handlers are notified now to avoid firewall
conflicts.]
*********************************************************************

RUNNING HANDLER [geerlingguy.docker : restart docker]
*************************************************************************************
*************************
changed: [129.0.0.2]

TASK [geerlingguy.docker : include_tasks]
*************************************************************************************
************************************
skipping: [129.0.0.2]
skipping: [129.0.0.3]

TASK [geerlingguy.docker : Get docker group info using getent.]
*************************************************************************************
****************
skipping: [129.0.0.2]
skipping: [129.0.0.3]

TASK [geerlingguy.docker : Check if there are any users to add to the docker
group.]
*********************************************************************************
skipping: [129.0.0.2]
skipping: [129.0.0.3]

TASK [geerlingguy.docker : include_tasks]
*************************************************************************************
************************************
skipping: [129.0.0.2]
skipping: [129.0.0.3]

TASK [docker_compose : Install Docker Compose]
*************************************************************************************
*********************************
included: /Users/i.nafikov/Study/iu-devsecops-course/lab-
02/ansible/roles/docker_compose/tasks/install_compose.yml for 129.0.0.2, 129.0.0.3

TASK [docker_compose : Update apt]
*************************************************************************************
*********************************************
```

```
changed: [129.0.0.2]
changed: [129.0.0.3]

TASK [docker_compose : Install docker-compose via apt]
*****************************************************************************************
*************************
ok: [129.0.0.3]
changed: [129.0.0.2]

TASK [docker_compose : Add default user to docker group]
*****************************************************************************************
************************
included: /Users/i.nafikov/Study/iu-devsecops-course/lab-
02/ansible/roles/docker_compose/tasks/add_user_to_group.yml for 129.0.0.2,
129.0.0.3

TASK [docker_compose : Add default user to docker group]
*****************************************************************************************
************************
changed: [129.0.0.2]
changed: [129.0.0.3

PLAY RECAP
*****************************************************************************************
*************************************************************
129.0.0.2                  : ok=20   changed=8    unreachable=0    failed=0
skipped=11   rescued=0    ignored=0
129.0.0.3                  : ok=19   changed=2    unreachable=0    failed=0
skipped=11   rescued=0    ignored=0
```

- You can notice that **`VM-prod`** (`129.0.0.3`) already had Docker and Compose because I made debugging of this playbook on it 😅
- Then I started `deploy_python_app` playbook to "test" Python application on "dev" environment.

```
ansible-playbook playbooks/deploy_python_app.yml
```

```
Terminal  Local
(ansible-env) i.nafikov@macbook-KN70WX2HPH ansible % ansible-playbook playbooks/deploy_python_app.yml

PLAY [Deploy Python Application] ***********************************************************************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************************************************************************
[WARNING]: Platform linux on host 129.0.0.2 is using the discovered Python interpreter at /usr/bin/python3.12, but future installation of another Python interpreter could change
the meaning of that path. See https://docs.ansible.com/ansible-core/2.18/reference_appendices/interpreter_discovery.html for more information.
ok: [129.0.0.2]

TASK [current_time_app : Check Docker Compose file exists] *********************************************************************************************************************************
ok: [129.0.0.2]

TASK [current_time_app : Remove Docker Compose services] ***********************************************************************************************************************************
skipping: [129.0.0.2]

TASK [current_time_app : Remove Docker Compose file] ***************************************************************************************************************************************
ok: [129.0.0.2]

TASK [current_time_app : Remove application's directory with it's content] *****************************************************************************************************************
ok: [129.0.0.2]

TASK [current_time_app : Create application directory] **************************************************************************************************************************************
changed: [129.0.0.2]

TASK [current_time_app : Create visits directory] *******************************************************************************************************************************************
changed: [129.0.0.2]

TASK [current_time_app : Create docker-compose.yml file from template] *********************************************************************************************************************
changed: [129.0.0.2]

RUNNING HANDLER [current_time_app : Restart Docker Compose] ********************************************************************************************************************************
changed: [129.0.0.2]

PLAY RECAP *********************************************************************************************************************************************************************************
129.0.0.2                  : ok=8    changed=4    unreachable=0    failed=0    skipped=1    rescued=0    ignored=0
```

- Finally, I run `deploy_kotlin_app` to "release" Kotlin service to "prod"

deploy_kotlin_app.yml

```yaml
- name: Deploy Kotlin Application
  hosts: prod
  roles:
    - name: current_time_app
      vars:
        app_name: app_kotlin
        docker_image_name: iskanred/app_kotlin
        docker_image_version: 1.0.0
        docker_container_name: app_kotlin_container
        internal_port: 8080
        external_port: 80
        web_app_full_stop: true
        web_app_full_wipe: false

```

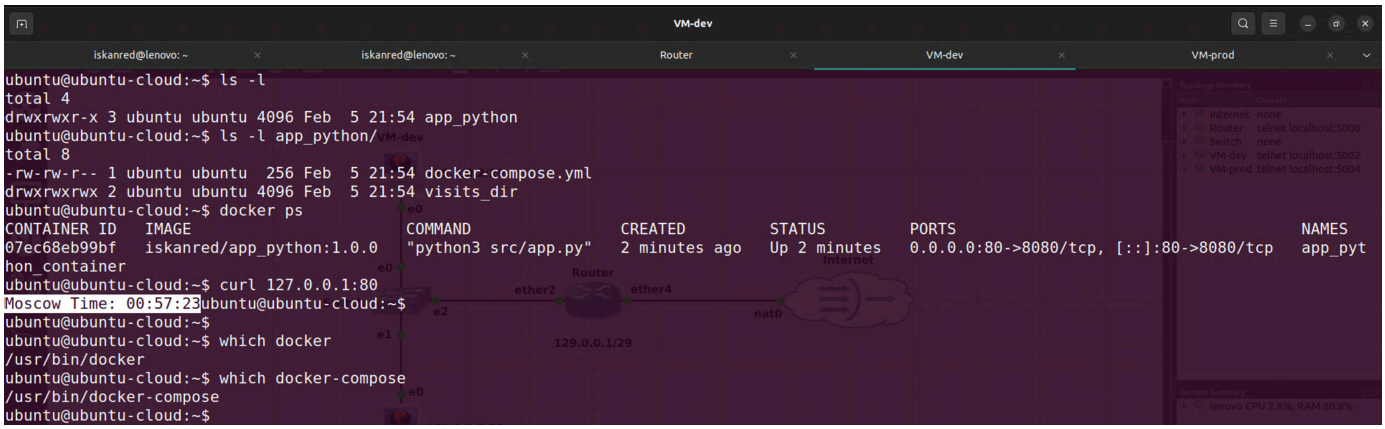
```
ansible-playbook playbooks/deploy_kotlin_app.yml
```

```
(ansible-env) i.nafikov@macbook-KN70WX2HPH ansible % ansible-playbook playbooks/deploy_kotlin_app.yml

PLAY [Deploy Kotlin Application] ***********************************************************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************************************************************
[WARNING]: Platform linux on host 129.0.0.3 is using the discovered Python interpreter at /usr/bin/python3.12, but future installation of another Python interpreter could change
the meaning of that path. See https://docs.ansible.com/ansible-core/2.18/reference_appendices/interpreter_discovery.html for more information.
ok: [129.0.0.3]

TASK [current_time_app : Check Docker Compose file exists] *********************************************************************************************************************************
ok: [129.0.0.3]

TASK [current_time_app : Remove Docker Compose services] ***********************************************************************************************************************************
skipping: [129.0.0.3]

TASK [current_time_app : Remove Docker Compose file] ***************************************************************************************************************************************
ok: [129.0.0.3]

TASK [current_time_app : Remove application's directory with it's content] *****************************************************************************************************************
skipping: [129.0.0.3]

TASK [current_time_app : Create application directory] **************************************************************************************************************************************
changed: [129.0.0.3]

TASK [current_time_app : Create visits directory] *******************************************************************************************************************************************
changed: [129.0.0.3]

TASK [current_time_app : Create docker-compose.yml file from template] *********************************************************************************************************************
changed: [129.0.0.3]

RUNNING HANDLER [current_time_app : Restart Docker Compose] ********************************************************************************************************************************
changed: [129.0.0.3]

PLAY RECAP *********************************************************************************************************************************************************************************
129.0.0.3                  : ok=7    changed=4    unreachable=0    failed=0    skipped=2    rescued=0    ignored=0
```

Results

Let's check the results of the deployment

- `VM-dev`
  - Software were installed, necessary directories were created, and the container was run



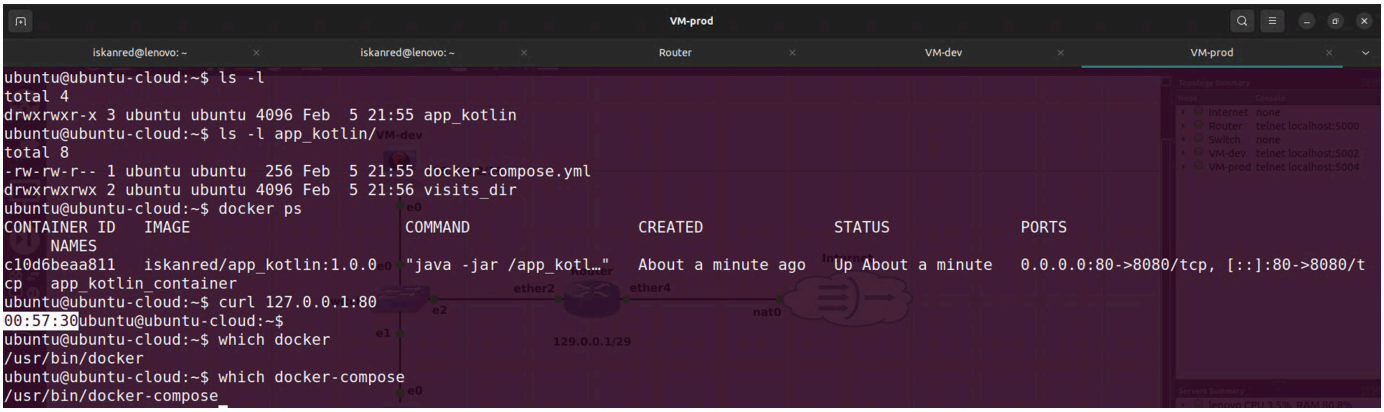
  - Finally, I can access the application from outside

Moscow Time: 01:00:36

10

- `VM-prod`

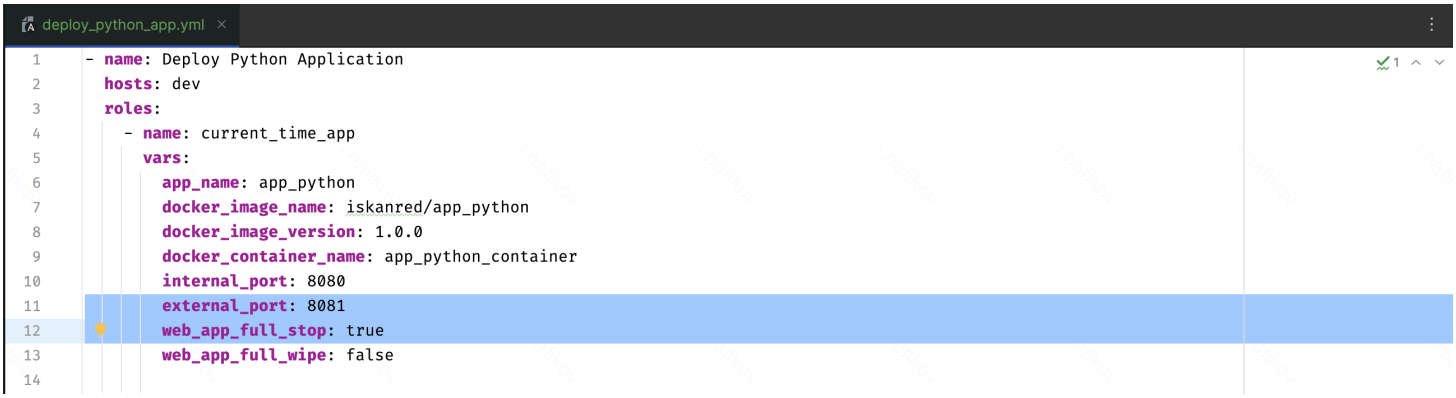
- Software were installed, necessary directories were created, and the container was run



- Finally, I can access the application from outside

Modifications

- Now let's enable flag `web_app_full_stop` to `true` and change port mapping from `8080 -> 80` to `8080 -> 8081` for `deploy_kotlin` playbook and run it again




```
Terminal  Local
(ansible-env) i.nafikov@macbook-KN70WX2HPH ansible % ansible-playbook playbooks/deploy_python_app.yml

PLAY [Deploy Python Application] ***********************************************************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************************************************************
[WARNING]: Platform linux on host 129.0.0.2 is using the discovered Python interpreter at /usr/bin/python3.12, but future installation of another Python interpreter could change
the meaning of that path. See https://docs.ansible.com/ansible-core/2.18/reference_appendices/interpreter_discovery.html for more information.
ok: [129.0.0.2]

TASK [current_time_app : Check Docker Compose file exists] *********************************************************************************************************************************
ok: [129.0.0.2]

TASK [current_time_app : Remove Docker Compose services] ***********************************************************************************************************************************
changed: [129.0.0.2]

TASK [current_time_app : Remove Docker Compose file] ***************************************************************************************************************************************
changed: [129.0.0.2]

TASK [current_time_app : Remove application's directory with it's content] *****************************************************************************************************************
skipping: [129.0.0.2]

TASK [current_time_app : Create application directory] *************************************************************************************************************************************
ok: [129.0.0.2]

TASK [current_time_app : Create visits directory] *******************************************************************************************************************************************
ok: [129.0.0.2]

TASK [current_time_app : Create docker-compose.yml file from template] *********************************************************************************************************************
changed: [129.0.0.2]

RUNNING HANDLER [current_time_app : Restart Docker Compose] ********************************************************************************************************************************
changed: [129.0.0.2]

PLAY RECAP *********************************************************************************************************************************************************************************
129.0.0.2                  : ok=8    changed=4    unreachable=0    failed=0    skipped=1    rescued=0    ignored=0
```

- We see that `Remove Docker Compose services` and `Remove Docker Compose file` tasks were now executed because `web_app_full_stop` is enabled and our service had been already up
- And now we can access our serve on **`VM-dev`** by 8081 port

- We see that the number of visits have not been reset and have been increased by one
- So now let's also enable `web_app_full_wipe` and check the results of running the playbook

```yaml
- name: Deploy Python Application
  hosts: dev
  roles:
    - name: current_time_app
      vars:
        app_name: app_python
        docker_image_name: iskanred/app_python
        docker_image_version: 1.0.0
        docker_container_name: app_python_container
        internal_port: 8080
        external_port: 8081
        web_app_full_stop: true
        web_app_full_wipe: true
```

```
(ansible-env) i.nafikov@macbook-KN70WX2HPH ansible % ansible-playbook playbooks/deploy_python_app.yml

PLAY [Deploy Python Application] ***********************************************************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************************************************************
[WARNING]: Platform linux on host 129.0.0.2 is using the discovered Python interpreter at /usr/bin/python3.12, but future installation of another Python interpreter could change
the meaning of that path. See https://docs.ansible.com/ansible-core/2.18/reference_appendices/interpreter_discovery.html for more information.
ok: [129.0.0.2]

TASK [current_time_app : Check Docker Compose file exists] *********************************************************************************************************************************
ok: [129.0.0.2]

TASK [current_time_app : Remove Docker Compose services] ***********************************************************************************************************************************
changed: [129.0.0.2]

TASK [current_time_app : Remove Docker Compose file] ***************************************************************************************************************************************
changed: [129.0.0.2]

TASK [current_time_app : Remove application's directory with it's content] *****************************************************************************************************************
changed: [129.0.0.2]

TASK [current_time_app : Create application directory] **************************************************************************************************************************************
changed: [129.0.0.2]

TASK [current_time_app : Create visits directory] *******************************************************************************************************************************************
changed: [129.0.0.2]

TASK [current_time_app : Create docker-compose.yml file from template] *********************************************************************************************************************
changed: [129.0.0.2]

RUNNING HANDLER [current_time_app : Restart Docker Compose] ********************************************************************************************************************************
changed: [129.0.0.2]

PLAY RECAP *********************************************************************************************************************************************************************************
129.0.0.2                  : ok=9    changed=7    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

(ansible-env) i.nafikov@macbook-KN70WX2HPH ansible %
```

- As we see the task `Remove application's directory with it's content` have also been executed
- And now we see 0 visits because this number was stored in the application's directory on **`VM-dev`** machine and it was reset

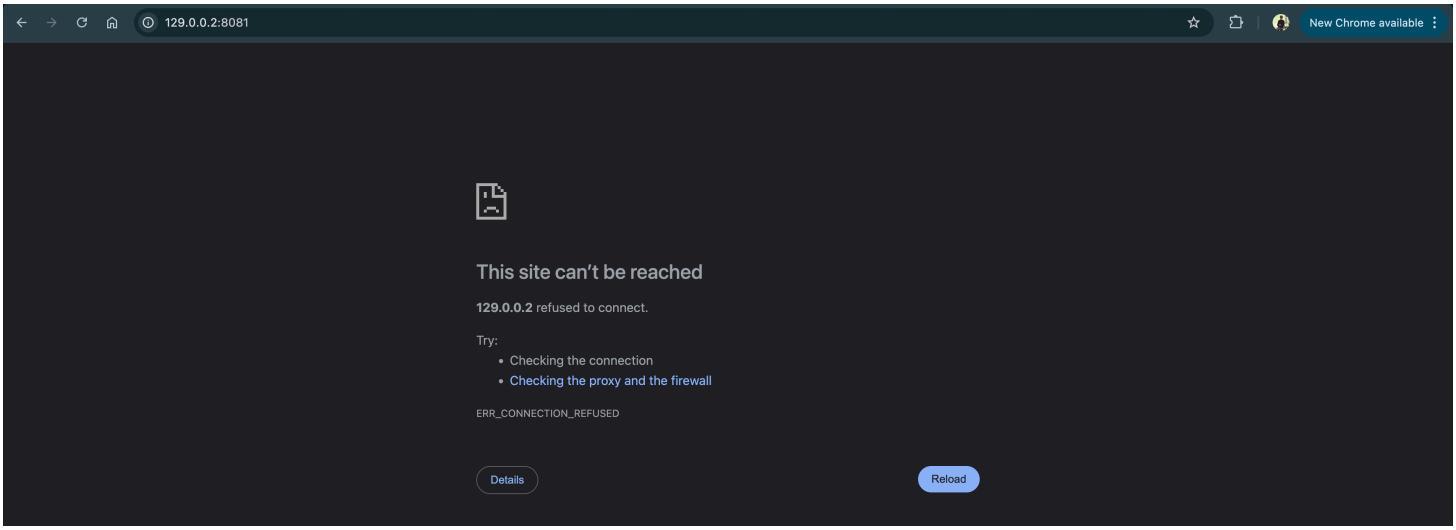
Finish

- Okay, now we can either stop (with saving visits info) or totally remove our application using Ansible tags. Let's remove it from **`VM-dev`** using `wipe-app` tag:

```
ansible-playbook playbooks/deploy_python_app.yml --tags=wipe-app
```

```
Terminal  Local
(ansible-env) i.nafikov@macbook-KN70WX2HPH ansible % ansible-playbook playbooks/deploy_python_app.yml --tags=wipe-app

PLAY [Deploy Python Application] ***********************************************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************************************************
[WARNING]: Platform linux on host 129.0.0.2 is using the discovered Python interpreter at /usr/bin/python3.12, but future installation of another Python interpreter could change
the meaning of that path. See https://docs.ansible.com/ansible-core/2.18/reference_appendices/interpreter_discovery.html for more information.
ok: [129.0.0.2]

TASK [current_time_app : Check Docker Compose file exists] *********************************************************************************************************************
ok: [129.0.0.2]

TASK [current_time_app : Remove Docker Compose services] ***********************************************************************************************************************
changed: [129.0.0.2]

TASK [current_time_app : Remove Docker Compose file] ***************************************************************************************************************************
changed: [129.0.0.2]

TASK [current_time_app : Remove application's directory with it's content] *****************************************************************************************************
changed: [129.0.0.2]

PLAY RECAP *********************************************************************************************************************************************************************
129.0.0.2                  : ok=5    changed=3    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

(ansible-env) i.nafikov@macbook-KN70WX2HPH ansible %
```

- We see that all the "remove" tasks have been executed but a new deployment have not been done
- And subsequently we cannot access our application anymore



2.

> Provide the link to git repository with all your labs configurations.

https://github.com/iskanred/iu-devsecops-course/tree/main/lab-02
