

# TP - AUTOMATISATION DU DÉPLOIEMENT D'UNE INFRASTRUCTURE AVEC ANSIBLE

05/03/2025

I) D'abord, je commence par crée 2 machine virtuelles sur Microsoft Azure (VM1 et VM2)

Pour rappel / prérequis:

**MobaXterm** pour se connecter aux machines en SSH grâce aux clés RSA

**VM1 (10.0.0.6) : Héberge Jenkins, Git et Ansible.**

**VM2 (10.0.0.7) : Héberge MySQL, Docker et Nginx.**

Ouvrir les ports nécessaires (22, 80, 443, 8080) dans le groupe de sécurité réseau.



Machines virtuelles

m2:Formation1 (m2:Formation1.onmicrosoft.com)

+ Créer Passer au mode classique Réervations Gérer la vue Actualiser Exporter au format CSV Ouvrir une requête Attribuer des étiquettes Démarrer Redémarrer Arrêter

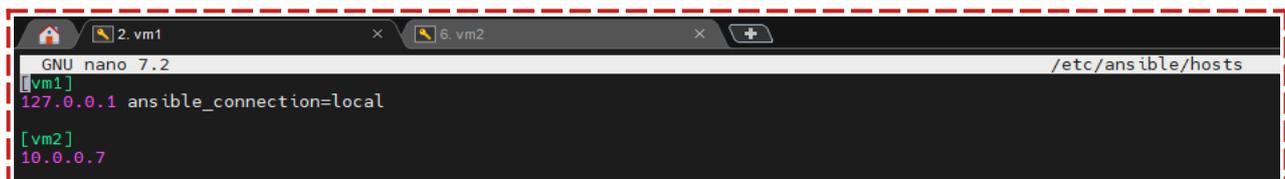
vm mehdi Abonnement égal à tout Type égal à tout Groupe de ressources égal à tout Emplacement égal à tout Ajouter un filtre

Affichage de 1 à 2 sur 2 enregistrements.

Nom	Abonnement	Groupe de ressources	Emplacement	État	Système d'exploitati...	Taille	Adresse IP publique
vm1-mehdi	Azure-Provisionning	PROJET-NMA	UK West	Exécution	Linux	Standard_B2s	20.162.85.205
vm2-mehdi	Azure-Provisionning	PROJET-NMA	UK West	Exécution	Linux	Standard_B2s	20.254.197.240

II) Sur **VM1**, installer **Ansible** et créer un fichier `inventory` pour référencer les VMs :

`sudo apt update && sudo apt install -y ansible`



```
GNU nano 7.2 /etc/ansible/hosts
[vm1]
127.0.0.1 ansible_connection=local
[vm2]
10.0.0.7
```

### III) Je peux tester les connexions entre mes machines

```
azureuser@vm1-mehdi:~$ ansible -i /etc/ansible/hosts all -m ping
127.0.0.1 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python3"
  },
  "changed": false,
  "ping": "pong"
}
10.0.0.7 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python3"
  },
  "changed": false,
  "ping": "pong"
}
```

### IV) Déploiement Automatisé avec Ansible

Création d'un fichier **playbook.yml** qui :

- Installe **Jenkins, Git et Ansible** sur **VM1**.
- Installe **MySQL, Docker et Nginx** sur **VM2**.
- Sécurise **MySQL** et configure **Nginx**.

```
azureuser@vm1-mehdi:~$ cat playbook.yml
- hosts: vm1
  become: yes
  tasks:
    - name: Mise à jour des paquets
      apt:
        update_cache: yes
    - name: Installer les dépendances pour ajouter des dépôts
      apt:
        name:
          - curl
          - gnupg
          - software-properties-common
        state: present
    - name: Ajouter le dépôt Jenkins
      shell: |
        curl -fsSL https://pkg.jenkins.io/debian/jenkins.io-2023.key | tee \
        /usr/share/keyrings/jenkins-keyring.asc > /dev/null
        echo "deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \
        https://pkg.jenkins.io/debian binary/" | tee \
        /etc/apt/sources.list.d/jenkins.list > /dev/null
      apt update
    - name: Installer Git, Ansible, OpenJDK et Jenkins
      apt:
        name:
          - git
          - ansible
          - openjdk-17-jdk
          - jenkins
        state: present
    - name: Démarrer et activer Jenkins
      service:
        name: jenkins
        state: started
        enabled: yes
- hosts: vm2
  become: yes
  tasks:
    - name: Mise à jour des paquets
      apt:
        update_cache: yes
    - name: Installer MySQL, Docker et Nginx
      apt:
        name:
          - mysql-server
          - docker.io
          - nginx
        state: present
    - name: Vérifier si MySQL est déjà installé
      stat:
        path: /etc/mysql/my.cnf
        register: mysql_installed
    - name: Sécuriser MySQL (définir un mot de passe root)
      debconf:
        name: mysql-server
        question: "mysql-server/root_password"
        value: "rootpassword"
        vtype: "password"
        when: mysql_installed.stat.exists == False
    - name: Sécuriser MySQL (confirmation mot de passe)
      debconf:
        name: mysql-server
        question: "mysql-server/root_password_again"
        value: "rootpassword"
        vtype: "password"
        when: mysql_installed.stat.exists == False
    - name: Démarrer et activer MySQL
      service:
        name: mysql
        state: started
        enabled: yes
    - name: Démarrer et activer Nginx
      service:
        name: nginx
        state: started
        enabled: yes
    - name: Remplacer la page d'accueil Nginx
      copy:
        content: "Bienvenue sur le serveur web de Mehdi pour le TP AUTOMATISATION DU DÉPLOIEMENT D'UNE INFRASTRUCTURE AVEC ANSIBLE !"
        dest: /var/www/html/index.html
```

## V) Exécution du playbook.yml + vérification du bon fonctionnement

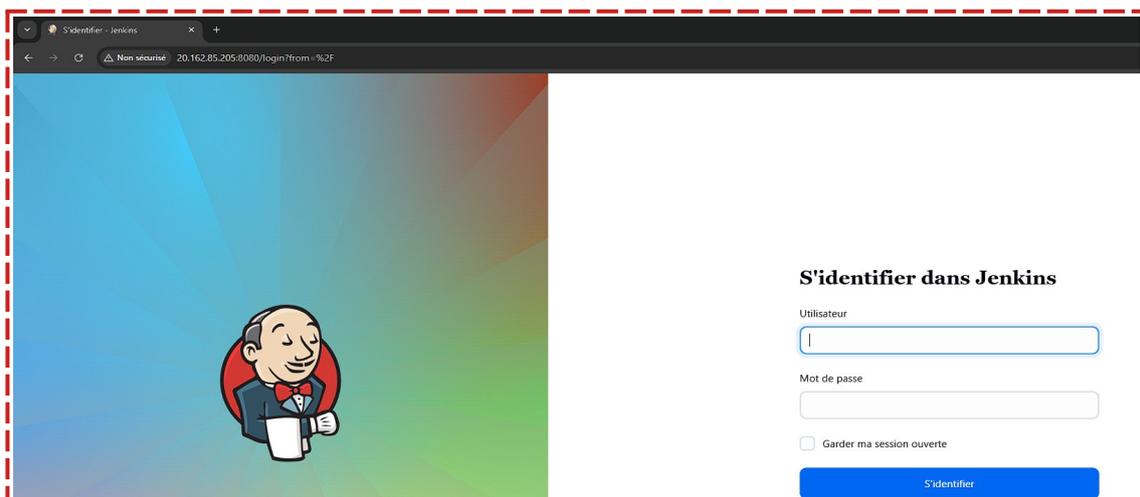
azureuser@vm1-mehdi:~\$ ansible-playbook -i /etc/ansible/hosts playbook.yml

```
azureuser@vm1-mehdi:~$ ansible-playbook -i /etc/ansible/hosts playbook.yml
PLAY [vm1] *************************************************************************************************************************************
TASK [Gathering Facts] *********************************************************************
ok: [127.0.0.1]
TASK [Mise à jour des paquets] *********************************************************************
changed: [127.0.0.1]
TASK [Installer les dépendances pour ajouter des dépôts] *********************************************************************
ok: [127.0.0.1]
TASK [Ajouter le dépôt Jenkins] *********************************************************************
changed: [127.0.0.1]
TASK [Installer Git, Ansible, OpenJDK et Jenkins] *********************************************************************
ok: [127.0.0.1]
TASK [Démarrer et activer Jenkins] *********************************************************************
ok: [127.0.0.1]
PLAY [vm2] *************************************************************************************************************************************
TASK [Gathering Facts] *********************************************************************
ok: [10.0.0.7]
TASK [Mise à jour des paquets] *********************************************************************
changed: [10.0.0.7]
TASK [Installer MySQL, Docker et Nginx] *********************************************************************
ok: [10.0.0.7]
TASK [Vérifier si MySQL est déjà installé] *********************************************************************
ok: [10.0.0.7]
TASK [Sécuriser MySQL (définir un mot de passe root)] *********************************************************************
skipping: [10.0.0.7]
TASK [Sécuriser MySQL (confirmation mot de passe)] *********************************************************************
skipping: [10.0.0.7]
TASK [Démarrer et activer MySQL] *********************************************************************
ok: [10.0.0.7]
TASK [Démarrer et activer Nginx] *********************************************************************
ok: [10.0.0.7]
TASK [Remplacer la page d'accueil Nginx] *********************************************************************
ok: [10.0.0.7]
PLAY RECAP *********************************************************************
10.0.0.7      : ok=7   changed=1   unreachable=0   failed=0   skipped=2   rescued=0   ignored=0
127.0.0.1    : ok=6   changed=2   unreachable=0   failed=0   skipped=0   rescued=0   ignored=0
```

Sur un navigateur avec l'ip Publique de ma VM2 :



Sur un navigateur avec l'ip Publique de ma VM1 au port 8080 :



Vérification de la bonne connexion de MySQL sur la VM2 :

```
azureuser@vm2-mehdi:~$ sudo mysql -u root -p -e "SHOW DATABASES;"
Enter password:
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| sys |
+-----+
```

VI) Ajout de la récupération du projet via Github sur le playbook.yml

```
- name: Cloner le projet depuis GitHub
  git:
    repo: "https://github.com/Meeehddiiii/simple_serveur_web.git"
    dest: "/home/azureuser/simple_serveur_web"
    version: main
```

VII) **Création du Dockerfile** : Un **Dockerfile** a été créé pour construire l'image Docker avec Nginx comme serveur web et le projet cloné comme contenu

```
- name: Créer le Dockerfile
  copy:
    dest: "/home/azureuser/simple_serveur_web/Dockerfile"
    content: |
      FROM nginx:latest
      COPY . /usr/share/nginx/html
      EXPOSE 80
      CMD ["nginx", "-g", "daemon off;"]
```

VIII) Création de l'image docker + supprimer si il existe un ancien conteneur nommée simple\_serveur\_weba

```
- name: Construire l'image Docker
  shell: |
    cd /home/azureuser/simple_serveur_web
    docker build -t simple_serveur_web .

- name: Supprimer l'ancien conteneur s'il existe
  shell: docker rm -f simple_serveur_web || true
```

IX) Exécution du conteneur Docker sur le port 8081

```
- name: Exécuter le conteneur Docker
  shell: |
    docker run -d -p 8081:80 --name simple_serveur_web simple_serveur_web
```

X) Création du fichier « credentials.yml » contenant les identifiants (username + token) Docker Hub

```
GNU nano 7.2 credentials.yml *
username: "meeeehddiiii"
password: "dcr_pat_N4*Xlw9*Nl2m*B*4v*h*hd*Q*8w"
```

XI) Chiffrer le fichier via la commande :

sudo ansible-vault encrypt credentials.yml

```
azureuser@vm1-mehdi:~$ sudo ansible-vault encrypt credentials.yml
New Vault password:
Confirm New Vault password:
Encryption successful
```

XII) Se connecter à Docker Hub + Taguer l'image + Pousser l'image sur Docker Hub

```
- name: Se connecter à Docker Hub
  shell: echo "{{ docker_hub_token }}" | docker login -u "{{ docker_hub_username }}" --password-stdin

- name: Taguer l'image Docker pour Docker Hub
  shell: docker tag simple_serveur_web {{ docker_hub_username }}/simple_serveur_web:latest

- name: Pousser l'image Docker sur Docker Hub
  shell: docker push {{ docker_hub_username }}/simple_serveur_web:latest
```

**Le fichier « playbook.yml » est disponible ici : [cliquez ici](#)**

XIII) On peut désormais lancer le playbook.yml grâce à la commande suivante

sudo ansible-playbook playbook.yml --ask-vault-pass

```
PLAY [vm1] *********************************************************************
TASK [Gathering Facts] *********************************************************
ok: [127.0.0.1]

TASK [Mise à jour des paquets] *************************************************
changed: [127.0.0.1]

TASK [Installer les dépendances pour ajouter des dépôts] ******************
ok: [127.0.0.1]

TASK [Ajouter le dépôt Jenkins] ************************************************
changed: [127.0.0.1]

TASK [Installer Git, Ansible, OpenJDK et Jenkins] **************************
ok: [127.0.0.1]

TASK [Démarrer et activer Jenkins] *********************************************
ok: [127.0.0.1]

PLAY [vm2] *********************************************************************
TASK [Gathering Facts] *********************************************************
ok: [10.0.0.7]

TASK [Mise à jour des paquets] *************************************************
changed: [10.0.0.7]

TASK [Installer MySQL, Docker et Nginx] *****
ok: [10.0.0.7]

TASK [Vérifier si MySQL est déjà installé] *****
ok: [10.0.0.7]

TASK [Sécuriser MySQL (définir un mot de passe root)] *****
skipping: [10.0.0.7]

TASK [Sécuriser MySQL (confirmation mot de passe)] *****
skipping: [10.0.0.7]

TASK [Démarrer et activer MySQL] *****
ok: [10.0.0.7]

TASK [Démarrer et activer Nginx] *****
ok: [10.0.0.7]

TASK [Cloner le projet depuis GitHub] *****
ok: [10.0.0.7]

TASK [Créer le Dockerfile] *****
ok: [10.0.0.7]

TASK [Construire l'image Docker] *****
changed: [10.0.0.7]

TASK [Supprimer l'ancien conteneur s'il existe] *****
changed: [10.0.0.7]

TASK [Exécuter le conteneur Docker] *****
changed: [10.0.0.7]

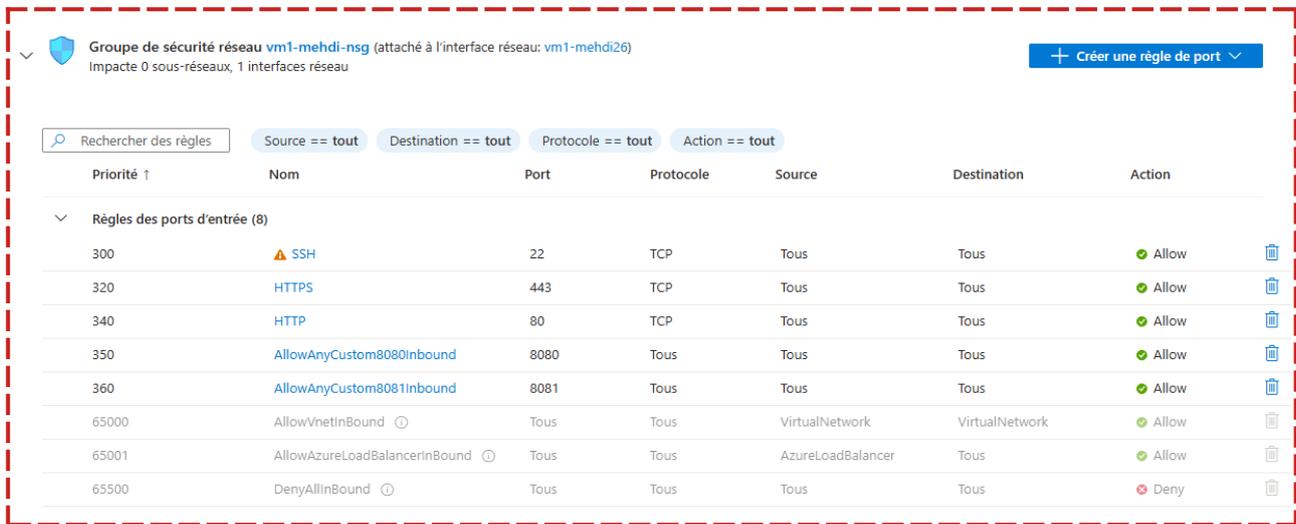
TASK [Se connecter à Docker Hub] *****
changed: [10.0.0.7]

TASK [Taguer l'image Docker pour Docker Hub] *****
changed: [10.0.0.7]

TASK [Pousser l'image Docker sur Docker Hub] *****
changed: [10.0.0.7]

PLAY RECAP *********************************************************************
10.0.0.7 : ok=18  changed=7  unreachable=0  failed=0  skipped=2  rescued=0  ignored=0
127.0.0.1 : ok=6   changed=2  unreachable=0  failed=0  skipped=0  rescued=0  ignored=0
```

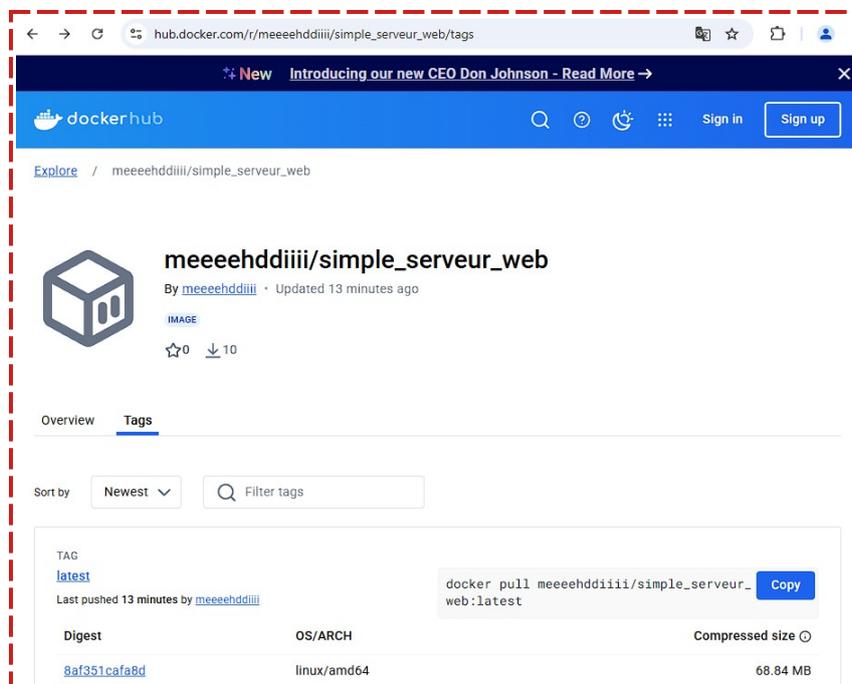
XIV) Désormais, j'ajoute une règle de **ports entrants** sur le **groupe de sécurité réseau (NSG)** dans Azure autorisant le trafic vers le port 8081 afin de pouvoir y accéder depuis un navigateur



Priorité ↑	Nom	Port	Protocole	Source	Destination	Action
300	SSH	22	TCP	Tous	Tous	Allow
320	HTTPS	443	TCP	Tous	Tous	Allow
340	HTTP	80	TCP	Tous	Tous	Allow
350	AllowAnyCustom8080Inbound	8080	Tous	Tous	Tous	Allow
360	AllowAnyCustom8081Inbound	8081	Tous	Tous	Tous	Allow
65000	AllowVnetInBound	Tous	Tous	VirtualNetwork	VirtualNetwork	Allow
65001	AllowAzureLoadBalancerInBound	Tous	Tous	AzureLoadBalancer	Tous	Allow
65500	DenyAllInBound	Tous	Tous	Tous	Tous	Deny



XIV) Vérification que le push vers Docker Hub a bien fonctionné directement sur le site de Docker Hub.



## XV) Configuration Pipeline Jenkins | Création du Jenkinsfile

```
GNU nano 7.2 Jenkinsfile
pipeline {
  agent any

  stages {
    stage('Checkout') {
      steps {
        // Cloner le dépôt GitHub en spécifiant la branche "main"
        git branch: 'main', url: 'https://github.com/Meeehddiiii/simple_serveur_web.git'
      }
    }

    stage('Vérification de la syntaxe HTML') {
      steps {
        // Vérification de la syntaxe du fichier index.html avec suppression de l'échec en cas d'avertissement
        script {
          def status = sh(script: 'tidy -e index.html', returnStatus: true)
          if (status != 0) {
            echo "Des avertissements ont été trouvés dans le fichier HTML."
          } else {
            echo "Aucun problème de syntaxe trouvé dans le fichier HTML."
          }
        }
      }
    }
  }
}
```

**Le Jenkinsfile que nous avons mis en place sert à automatiser un processus de validation de la syntaxe HTML de ton fichier index.html sur chaque commit poussé dans le dépôt GitHub.**

**Il permet de détecter les erreurs majeures et mineures dans la syntaxe du fichier HTML en utilisant l'outil tidy.**

### En résumé :

- **Les erreurs graves (majeures)**, comme les erreurs de balises ou d'autres violations importantes de la syntaxe HTML, seront détectées et rapportées.
- **Les avertissements (mineurs)**, comme l'absence de la déclaration `<!DOCTYPE>`, seront également signalés, mais ne feront pas échouer le pipeline à moins que ce comportement ne soit configuré autrement.

## XVI) Configuration Pipeline Jenkins | Création de l'item

Tableau de bord > Tous > test-git-repo-pipe > Configuration

Configurer

- Général
- Triggers
- Pipeline
- Advanced

Definition

Pipeline script from SCM

SCM ?

Git

Repositories ?

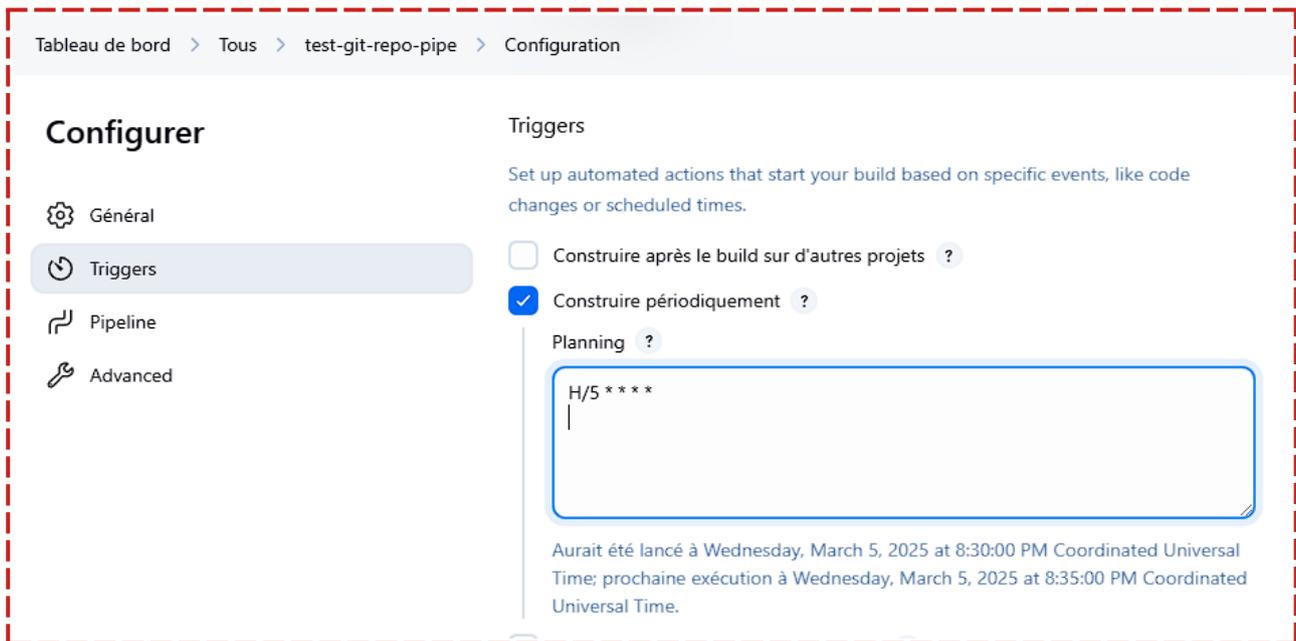
Repository URL ?

https://github.com/Meeehddiiii/simple\_serveur\_web

Credentials ?

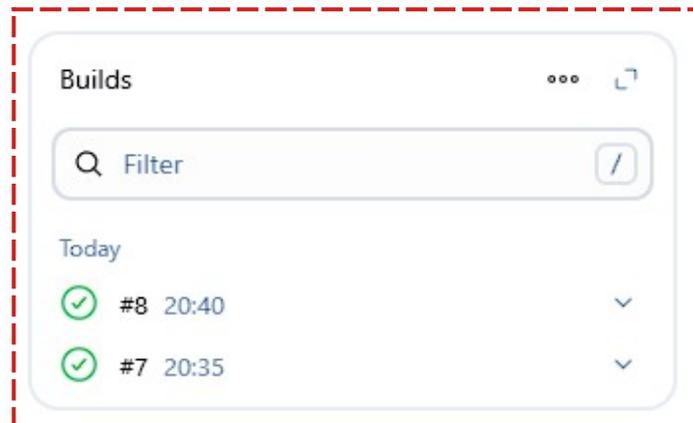
Meeehddiiii/\*\*\*\*\*

Je configure un trigger afin qu'un build soit créé toutes les 5 minutes.



The screenshot shows the configuration interface for triggers. The breadcrumb navigation at the top reads: "Tableau de bord > Tous > test-git-repo-pipe > Configuration". On the left, under the heading "Configurer", there is a sidebar menu with four items: "Général" (with a gear icon), "Triggers" (with a clock icon and highlighted), "Pipeline" (with a flowchart icon), and "Advanced" (with a key icon). The main content area is titled "Triggers" and includes the instruction: "Set up automated actions that start your build based on specific events, like code changes or scheduled times." There are two options: "Construire après le build sur d'autres projets" (unchecked) and "Construire périodiquement" (checked). Below the checked option is a "Planning" field containing the cron expression "H/5 \* \* \* \*". A note below the field states: "Aurait été lancé à Wednesday, March 5, 2025 at 8:30:00 PM Coordinated Universal Time; prochaine exécution à Wednesday, March 5, 2025 at 8:35:00 PM Coordinated Universal Time."

Désormais, je peux apercevoir qu'un build est créé toutes les 5 minutes



The screenshot shows a "Builds" list with a search filter. The list is titled "Builds" and has a search bar with the text "Filter" and a search icon. Below the search bar, the word "Today" is displayed. There are two build entries, each with a green checkmark icon, a build number, and a time: "#8 20:40" and "#7 20:35". Each entry has a dropdown arrow to its right.

# Fiche 4: Préparer un environnement de test

06/03/2025

## I) Configuration de l'Agent Jenkins

### Pour rappel / prérequis:

Pour permettre à **VM1** (serveur Jenkins) de se connecter à **VM2** (l'agent), une connexion SSH sécurisée a été établie entre les deux machines.

### Objectif :

Mettre en place un agent Jenkins sur la machine virtuelle **VM2**, en utilisant **VM1** (le serveur Jenkins) pour gérer les tâches à distance.

## II) Installation et Préparation sur VM2 (Agent Jenkins)

Avant de configurer l'agent Jenkins, il fallait s'assurer que **VM2** était prête pour recevoir les tâches Jenkins. Nécessitant Java pour son fonctionnement, J'installe **OpenJDK 11** :

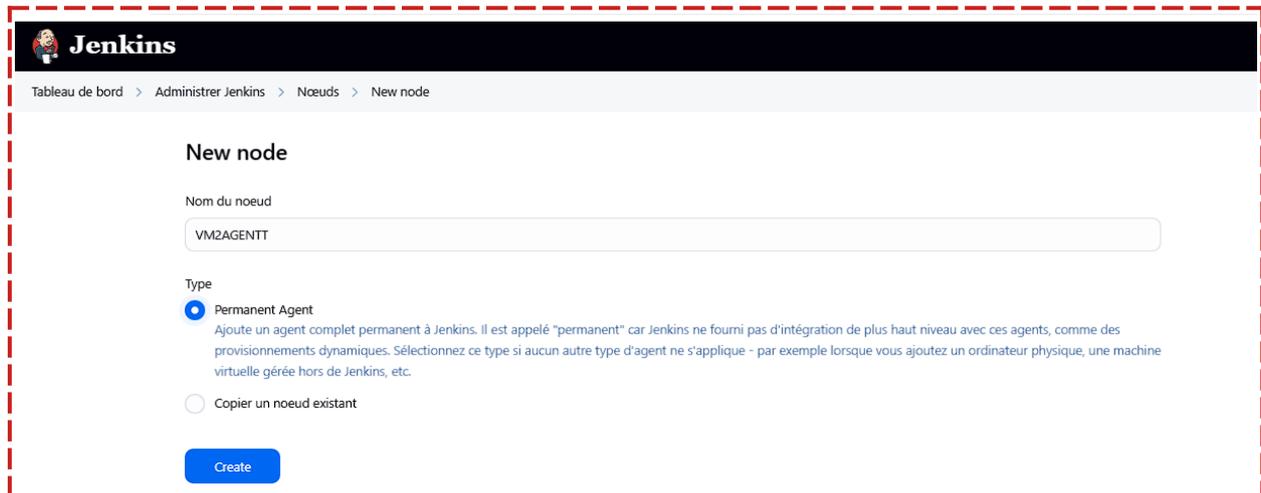
```
sudo apt update  
sudo apt install openjdk-11-jdk
```

Vérifier que Java est bien installer :

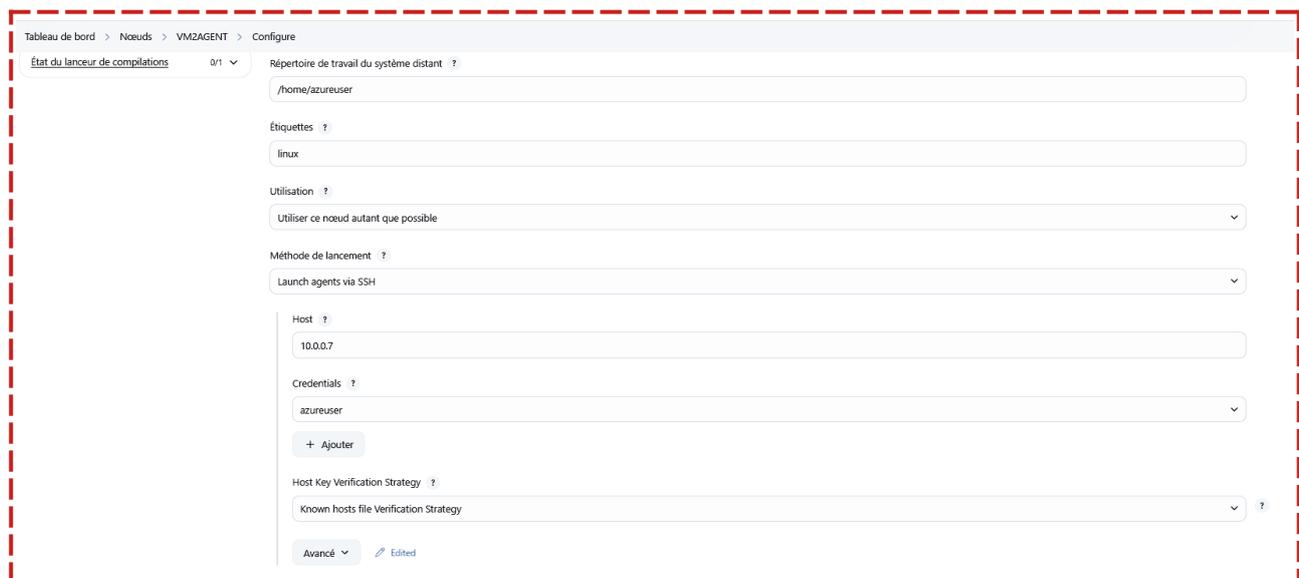
```
azureuser@vm2-mehdi:~$ java --version  
openjdk 21.0.6 2025-01-21  
OpenJDK Runtime Environment (build 21.0.6+7-Ubuntu-124.04.1)  
OpenJDK 64-Bit Server VM (build 21.0.6+7-Ubuntu-124.04.1, mixed mode, sharing)
```

## II) Configuration de l'Agent Jenkins

Depuis l'interface web de Jenkins sur **VM1**, un nouvel agent a été ajouté dans **"Manage Jenkins" > "Manage Nodes and Clouds"**



The screenshot shows the Jenkins 'New node' configuration page. The breadcrumb trail is 'Tableau de bord > Administrer Jenkins > Nœuds > New node'. The page title is 'New node'. There is a text input field for 'Nom du noeud' containing 'VM2AGENTT'. Under the 'Type' section, the 'Permanent Agent' option is selected with a radio button. A description for 'Permanent Agent' is provided: 'Ajoute un agent complet permanent à Jenkins. Il est appelé "permanent" car Jenkins ne fournit pas d'intégration de plus haut niveau avec ces agents, comme des provisionnements dynamiques. Sélectionnez ce type si aucun autre type d'agent ne s'applique - par exemple lorsque vous ajoutez un ordinateur physique, une machine virtuelle gérée hors de Jenkins, etc.' The 'Copier un noeud existant' option is unselected. A blue 'Create' button is at the bottom.



The screenshot shows the Jenkins 'Configure' page for the 'VM2AGENT' node. The breadcrumb trail is 'Tableau de bord > Nœuds > VM2AGENT > Configure'. The page title is 'Configure'. There is a dropdown menu for 'État du lanceur de compilations' set to '0/1'. The 'Répertoire de travail du système distant' field contains '/home/azureuser'. The 'Étiquettes' field contains 'linux'. The 'Utilisation' dropdown is set to 'Utiliser ce noeud autant que possible'. The 'Méthode de lancement' dropdown is set to 'Launch agents via SSH'. The 'Host' field contains '10.0.0.7'. The 'Credentials' dropdown is set to 'azureuser'. There is a '+ Ajouter' button below the credentials field. The 'Host Key Verification Strategy' dropdown is set to 'Known hosts file Verification Strategy'. At the bottom, there is an 'Avancé' dropdown and an 'Edited' link.

### Résumé de la Configuration de l'Agent Jenkins :

- **Nom de l'Agent** : VM2Agent
- **Méthode de Lancement** : Launch agent by SSH
- **Host** : 10 . 0 . 0 . 7 (IP de l'agent VM2)
- **Credentials** : **Username** : azureuser (utilisateur sur VM2) + clé SSH permettre l'authentification SSH entre VM1 et VM2
- **Répertoire de travail du système distant** : /home/azureuser (répertoire d'accueil de VM2 où les builds Jenkins seront exécutés)

### III) Vérification et Mise en Service de l'Agent

Dans le tableau de bord → Nœuds → VM2-Agent → Journal, on peut voir que l'agent est bien connecté.

```
<===[JENKINS REMOTING CAPACITY]===>channel started
Remoting version: 3283.v92c105e0f819
Launcher: SSHLauncher
Communication Protocol: Standard in/out
This is a Unix agent
Agent successfully connected and online
```

### IV) Lancer un build grâce à l'agent sur la VM2

Je crée un nouveau job Jenkins pour exécuter des commandes Shell sur VM2 via l'agent configuré

Je coche la case suivante et renseigne le nom de mon agent ou mon label

Restreindre où le projet peut être exécuté ?

Expression ?

VM2AGENT

Label VM2AGENT respecte 1 node. Les droits ou autres restrictions apportées par des plugins peuvent réduire cette liste.

Au niveau de l'étape de build, j'exécute un script shell à titre d'exemple

Étapes du build

Automate your build process with ordered tasks like code compilation, testing, and deployment.

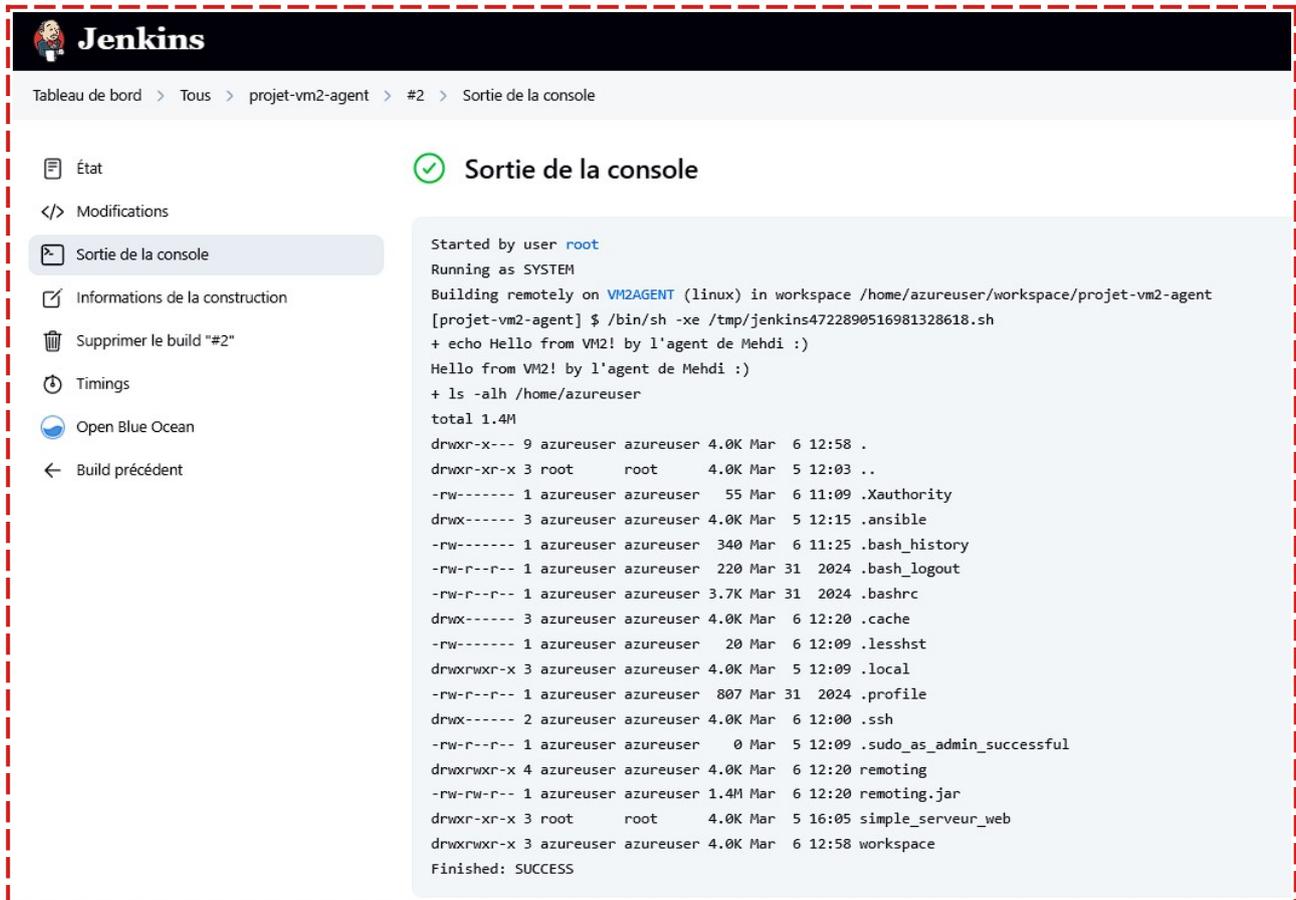
☰ Exécuter un script shell ?

Commande

[Voir la liste des variables d'environnement disponibles](#)

```
echo "Hello from VM2! by l'agent de Mehdi :)" |
ls -alh /home/azureuser
```

Puis je lance le build et observe le résultat sur ma machine VM2 qui contient l'agent Jenkins :



The screenshot shows the Jenkins web interface. The breadcrumb navigation is: `Tableau de bord > Tous > projet-vm2-agent > #2 > Sortie de la console`. The left sidebar contains several menu items: `État`, `Modifications`, `Sortie de la console` (highlighted), `Informations de la construction`, `Supprimer le build "#2"`, `Timings`, `Open Blue Ocean`, and `Build précédent`. The main content area displays the console output for the build, which is titled `Sortie de la console` with a green checkmark icon. The output text is as follows:

```
Started by user root
Running as SYSTEM
Building remotely on VM2AGENT (linux) in workspace /home/azureuser/workspace/projet-vm2-agent
[projet-vm2-agent] $ /bin/sh -xe /tmp/jenkins4722890516981328618.sh
+ echo Hello from VM2! by l'agent de Mehdi :)
Hello from VM2! by l'agent de Mehdi :)
+ ls -alh /home/azureuser
total 1.4M
drwxr-x--- 9 azureuser azureuser 4.0K Mar  6 12:58 .
drwxr-xr-x 3 root      root      4.0K Mar  5 12:03 ..
-rw----- 1 azureuser azureuser  55 Mar  6 11:09 .Xauthority
drwx----- 3 azureuser azureuser 4.0K Mar  5 12:15 .ansible
-rw----- 1 azureuser azureuser  340 Mar  6 11:25 .bash_history
-rw-r--r-- 1 azureuser azureuser  220 Mar 31  2024 .bash_logout
-rw-r--r-- 1 azureuser azureuser  3.7K Mar 31  2024 .bashrc
drwx----- 3 azureuser azureuser 4.0K Mar  6 12:20 .cache
-rw----- 1 azureuser azureuser   20 Mar  6 12:09 .lessht
drwxrwxr-x 3 azureuser azureuser 4.0K Mar  5 12:09 .local
-rw-r--r-- 1 azureuser azureuser  807 Mar 31  2024 .profile
drwx----- 2 azureuser azureuser 4.0K Mar  6 12:00 .ssh
-rw-r--r-- 1 azureuser azureuser    0 Mar  5 12:09 .sudo_as_admin_successful
drwxrwxr-x 4 azureuser azureuser 4.0K Mar  6 12:20 remoting
-rw-rw-r-- 1 azureuser azureuser 1.4M Mar  6 12:20 remoting.jar
drwxr-xr-x 3 root      root      4.0K Mar  5 16:05 simple_serveur_web
drwxrwxr-x 3 azureuser azureuser 4.0K Mar  6 12:58 workspace
Finished: SUCCESS
```