**MODULE : Virtualisation et Cloud Computing**

**Travaux Pratiques :**

**Déploiement d’une Infrastructure Multi-Machines avec Vagrant et Bash**

**GROUPE N°9**

**ENSEIGNANT :**

**KABORE Jean**

**KAGAMBEGA Boukary**

**KIENTEGA Francis**

**PARE K Léandre Bénilde**

**Dr SOMDA**

# **SOMMAIRE**

[**SOMMAIRE** 2](#_Toc195653864)

[**Listes des figures** 3](#_Toc195653865)

[**Introduction** 4](#_Toc195653866)

[**1. Prérequis et environnement** 4](#_Toc195653867)

[**2. Architecture de l’infrastructure** 5](#_Toc195653868)

[**3. Déploiement et Configuration** 7](#_Toc195653869)

[**Conclusion** 13](#_Toc195653870)

# **Listes des figures**

[**Figure 1 : Structure du TP** 5](#_Toc195653613)

[**Figure 2 : Schéma de l’infrastructure** 6](#_Toc195653614)

[**Figure 3 : Initialisation de vagrant** 7](#_Toc195653615)

[**Figure 4 : Creation des scripts** 7](#_Toc195653616)

[**Figure 5 : Virtual box avant VMs** 8](#_Toc195653617)

[**Figure 6 : Lancement de l’infrastructure** 8](#_Toc195653618)

[**Figure 7 : Virtual box Après VMs** 9](#_Toc195653619)

[**Figure 8 : Vérification la connectivité depuis le client en utilisant curl** 9](#_Toc195653620)

[**Figure 9 : DB master** 11](#_Toc195653621)

[**Figure 10 : DB slave** 12](#_Toc195653622)

# **Introduction**

Dans le cadre de ce travail pratique, nous avons été amenés à mettre en place une infrastructure distribuée à l’aide de l’outil Vagrant, associé à VirtualBox pour la virtualisation. L’objectif principal de ce TP est de simuler une architecture système complète comprenant un serveur de base de données, plusieurs serveurs web, un load balancer, un serveur de supervision, et un client.

Cette infrastructure repose sur une application web développée avec le framework Django, déployée sur deux serveurs web, et connectée à une base de données MySQL centralisée. Le choix de Django permet de bénéficier d’un cadre structuré pour le développement web, avec une interface d’administration, un ORM performant, et une architecture MVC robuste.

Le tout est conçu pour être facilement déployable et reproductible grâce à Vagrant et des scripts de provisioning automatisés.

**Objectifs du TP**

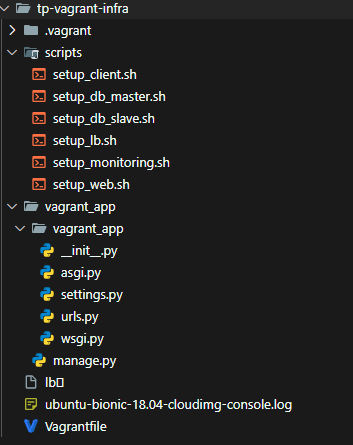
* Comprendre le fonctionnement de Vagrant et l’utiliser pour déployer plusieurs machines virtuelles automatiquement.
* Simuler une architecture distribuée réaliste composée de plusieurs rôles serveur.
* Automatiser le déploiement des services à l’aide de scripts de provisioning Bash.
* Déployer une application Django connectée à une base de données MySQL.
* Mettre en place un load balancer avec Nginx pour distribuer la charge entre deux serveurs web.
* Superviser l’état des serveurs à l’aide d’un outil de monitoring (comme Netdata).
* Vérifier le bon fonctionnement de l’infrastructure depuis une machine cliente.

# **1. Prérequis et environnement**

Pour réaliser ce TP, l’environnement de travail est basé sur une machine hôte équipée des outils suivants :

* **Outils nécessaires**
* Système d’exploitation hôte : Ubuntu 18.04
* Vagrant : version 2.4.3\_windows\_amd64
* VirtualBox : version-7.1.6-167084-Win
* Éditeur de texte / code : Visual Studio Code
* Python version 3.12.0
* **Structure du projet**

Le TP est organisé dans un dossier racine appelé tp-vagrant-infra, contenant :



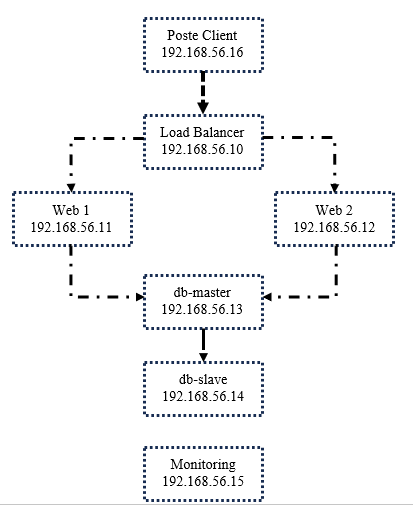
**Figure 1 : Structure du TP**

# **2. Architecture de l’infrastructure**

L’infrastructure mise en place se compose de sept machines virtuelles, chacune jouant un rôle spécifique au sein de l’architecture. Toutes les machines sont connectées en réseau privé via l’interface 192.168.56.0/24.

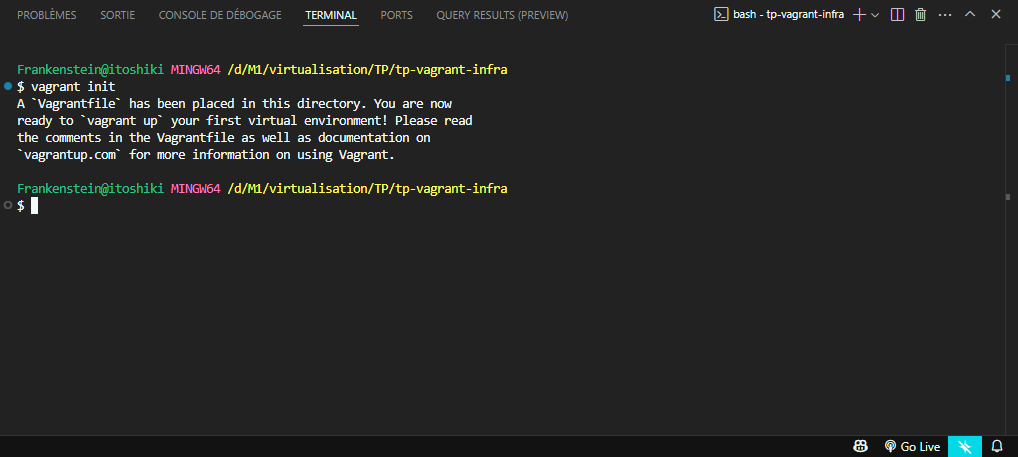
**Tableau I : Composition de l’infrastructure**

|  |  |  |  |
| --- | --- | --- | --- |
| Nom de laVM | Adresse IP | Rôle | Script de configuration |
| lb | 192.168.56.10 | Load Balancer (Nginx) | setup\_lb.sh |
| web1 | 192.168.56.11 | Serveur Web (Django) | setup\_web.sh |
| web2 | 192.168.56.12 | Serveur Web (Django) | setup\_web.sh |
| db-master | 192.168.56.13 | Base de données principale | setup\_db\_master.sh |
| db-slave | 192.168.56.14 | Réplication base de données | setup\_db\_slave.sh |
| monitoring | 192.168.56.15 | Supervision (Netdata) | setup\_monitoring.sh |
| client | 192.168.56.16 | Poste client (tests) | *(aucun script spécifique)* |

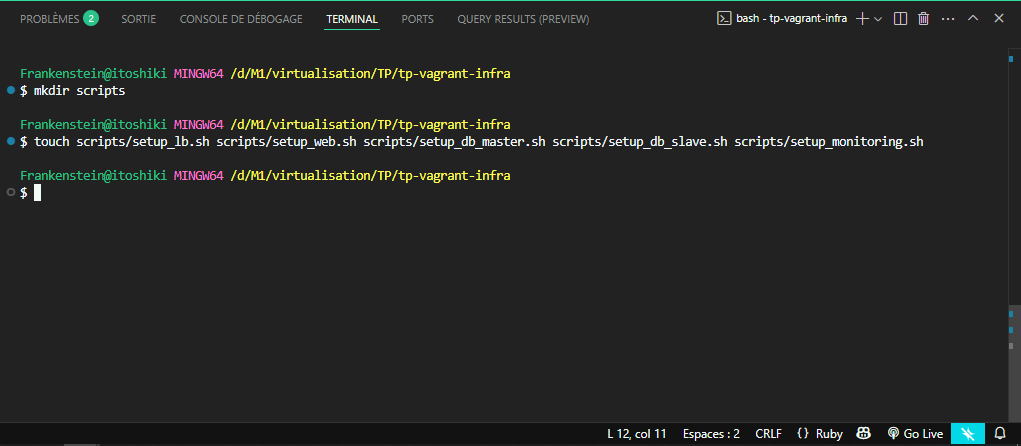


**Figure 2 : Schéma de l’infrastructure**

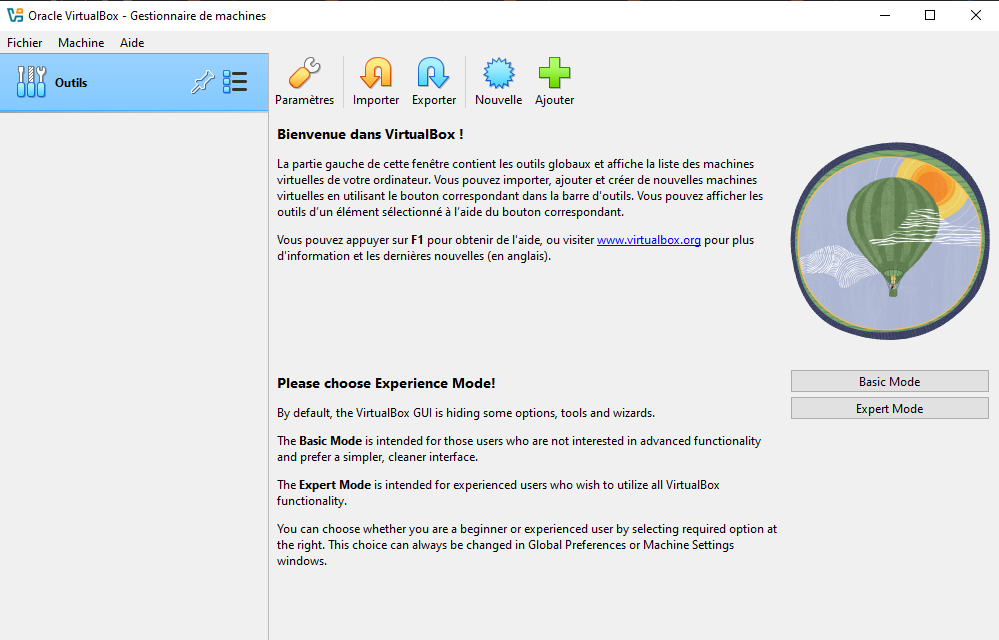
# **3. Déploiement et Configuration**



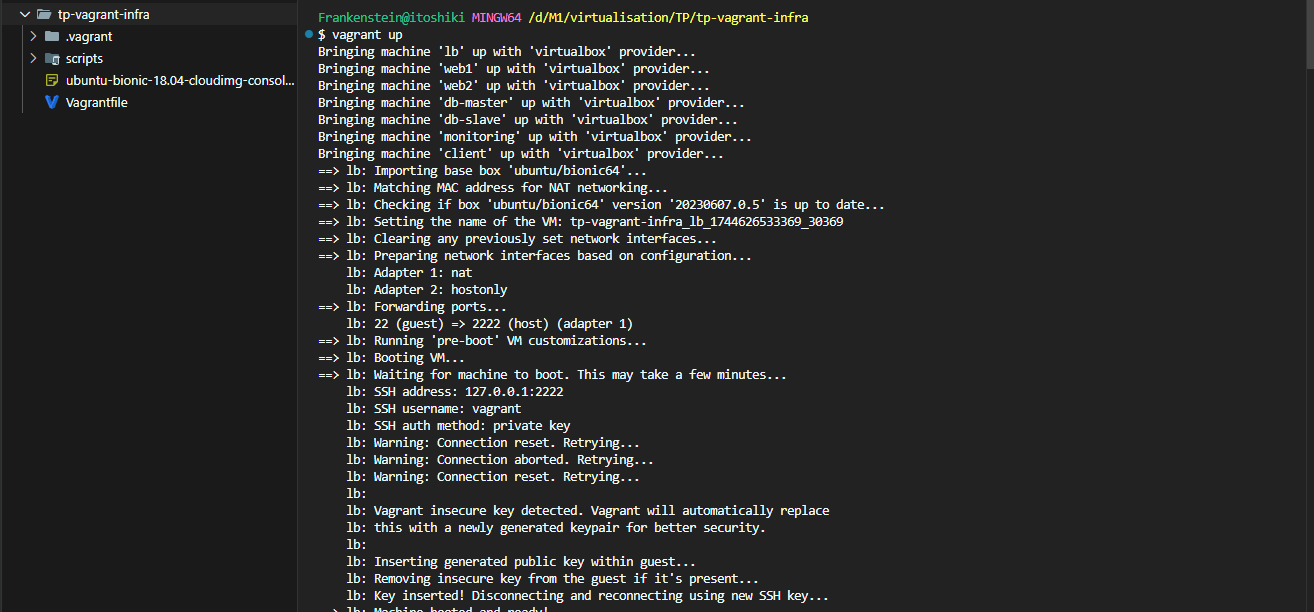
**Figure 3 : Initialisation de vagrant**



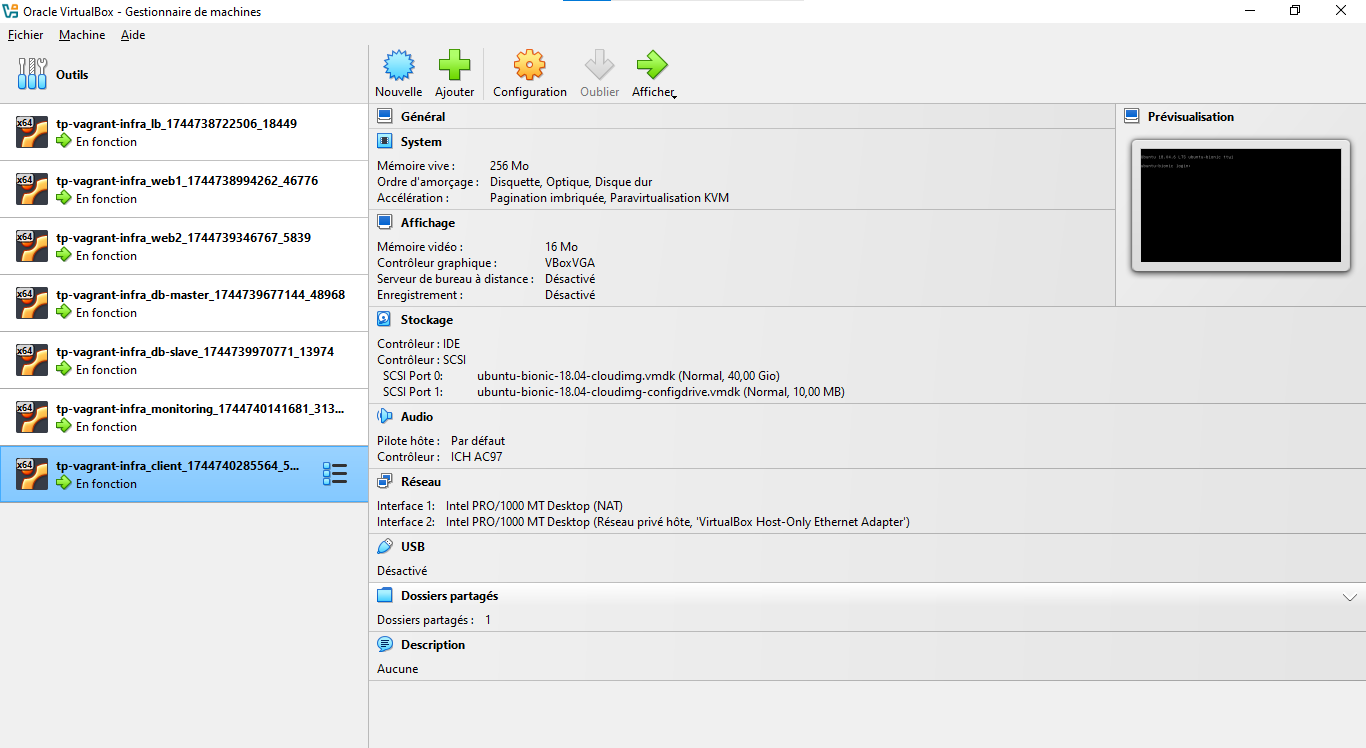
**Figure 4 : Creation des scripts**



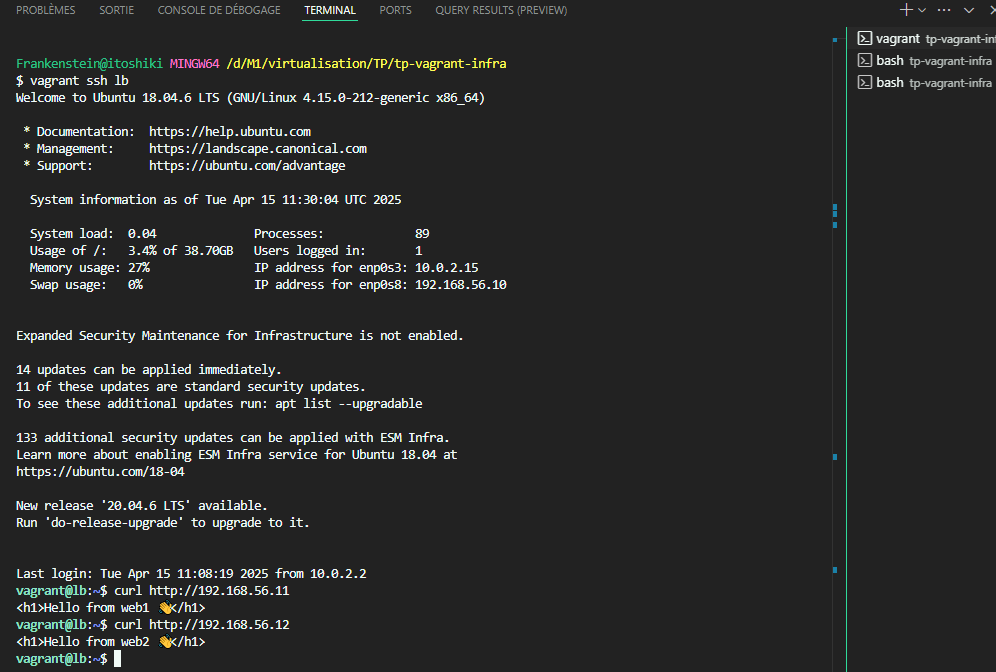
**Figure 5 : Virtual box avant VMs**



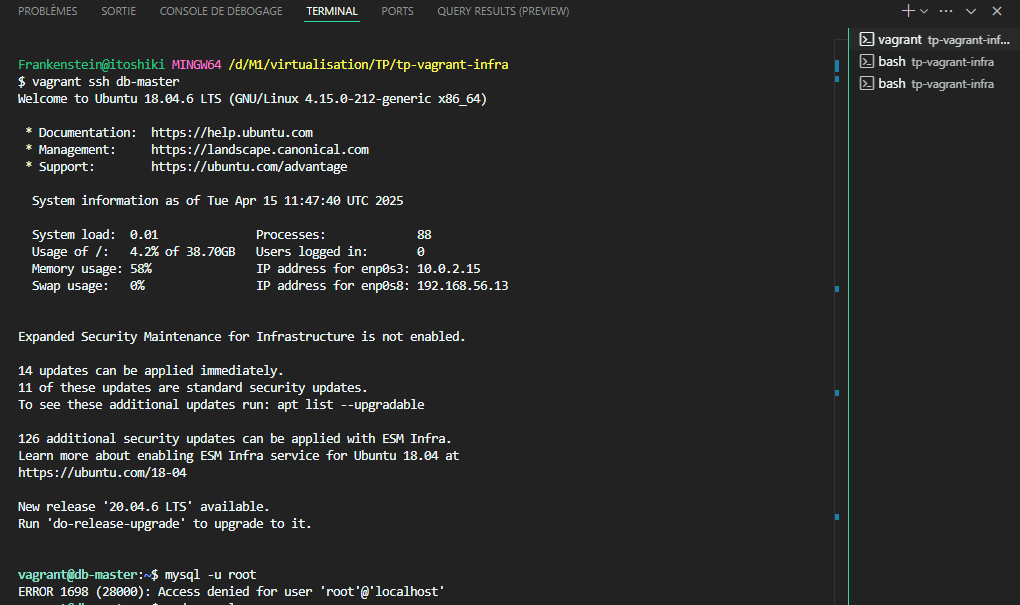
**Figure 6 : Lancement de l’infrastructure**

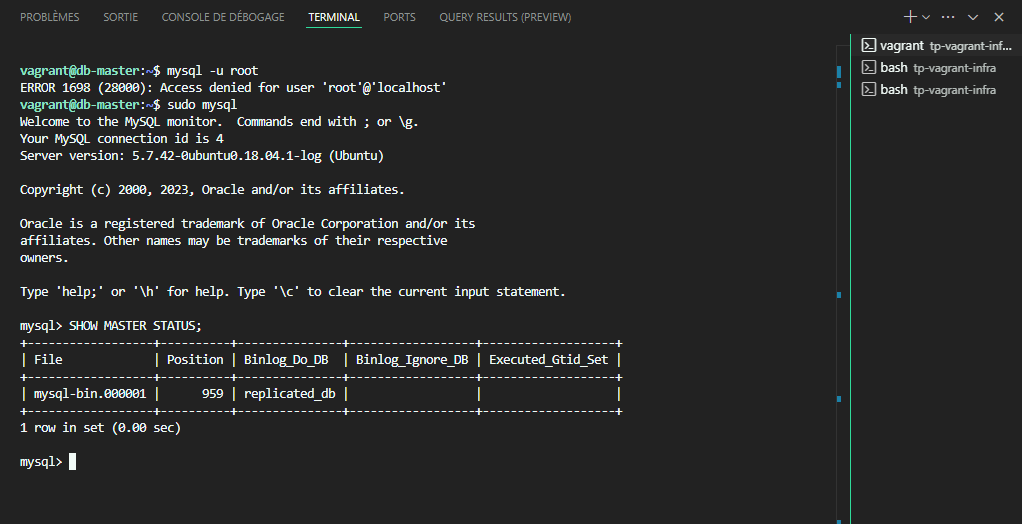


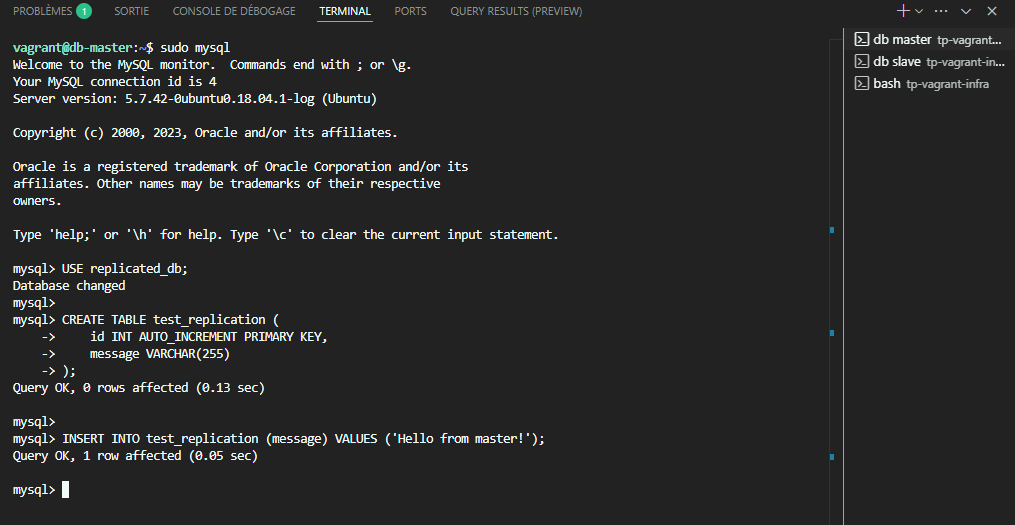
**Figure 7 : Virtual box Après VMs**



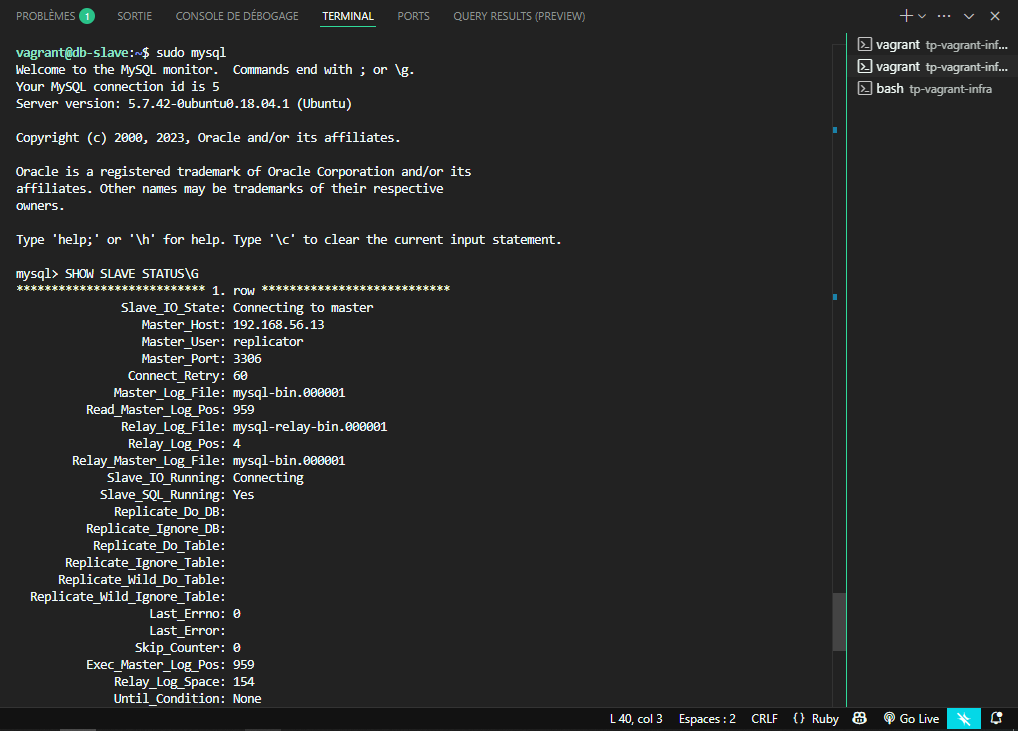
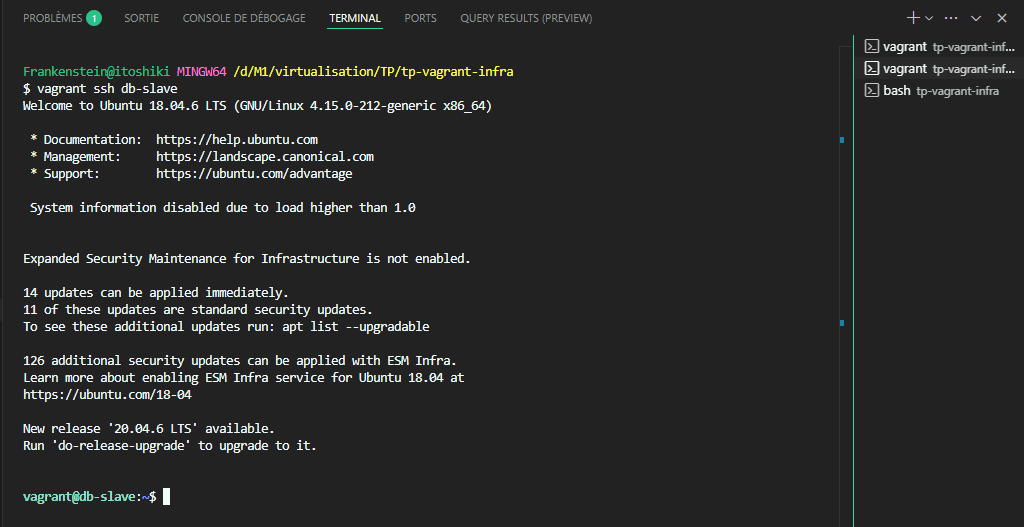
**Figure 8 : Vérification la connectivité depuis le client en utilisant curl**







**Figure 9 : DB master**



**Figure 10 : DB slave**

# **Conclusion**

Ce TP a permis de mettre en pratique plusieurs compétences fondamentales en administration système et réseau : déploiement automatisé, répartition de charge, réplication de base de données, supervision et test de résilience.

L'automatisation avec Vagrant et Bash simplifie grandement le déploiement d'environnements complexes, et la supervision permet d'assurer la bonne santé du système en temps réel.

Lien du dépôt : https://github.com/Korosenei/TP-Groupe-09/