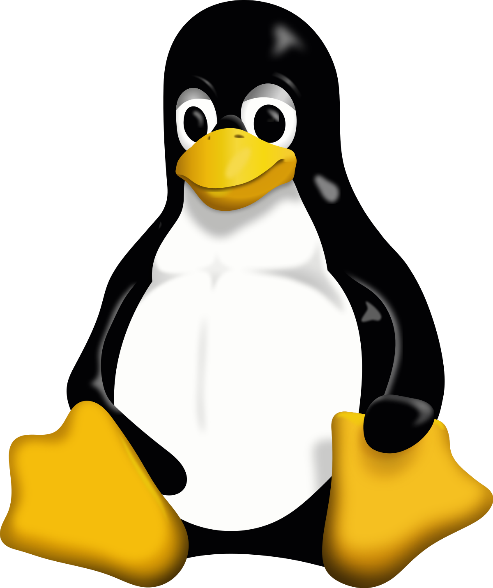
**Guide de l’utilisateur.**



Projet de Linux/UNIX

Projet LINUX Matrix Collège.

Équipe : Kevin Clovel, Bruno de Santana, Mario Camelo.

Prof : William Herrera.

Date : 24/01/2022.

INDICE

[INSTALLER DEUX SYSTÈMES D’EXPLOITATION LINUX SUR ORACLE VM. 3](#_Toc93756604)

[PRÉREQUIS : 3](#_Toc93756605)

[SERVEUR : 3](#_Toc93756606)

[CLIENT : 20](#_Toc93756607)

[COMMUNIQUER LES DEUX MACHINES (SERVEUR ET CLIENT) À L’AIDE DE DEUX ADAPTATEURS RÉSEAU. 23](#_Toc93756608)

[CRÉER L’UTILISATEUR « PROJET ». 28](#_Toc93756609)

[GÉNÉRER UNE CLÉ SSH SUR LE SERVEUR POUR L’UTILISATEUR DU PROJET. 29](#_Toc93756610)

[Copier la clé publique sur le client. 31](#_Toc93756611)

[Valider qu`il y a bien communication entre le serveur et le client en transmettant le fichier /home/projet/test.txt du client au serveur. 32](#_Toc93756612)

[LE SCRIPT. 34](#_Toc93756613)

[PRÉREQUIS: 34](#_Toc93756614)

[MENU SUIVANT SOUS LINUX. 36](#_Toc93756615)

[LA SCRUCTURE DU SCRIPT. 40](#_Toc93756616)

[LEXIQUE DES CODES. 40](#_Toc93756617)

[MAIN.SH 40](#_Toc93756618)

[CREATE\_FILE.SH 44](#_Toc93756619)

[VALIDE.SH 44](#_Toc93756620)

[CKSUM.SH 45](#_Toc93756621)

[SEND.SH 46](#_Toc93756622)

# INSTALLER DEUX SYSTÈMES D’EXPLOITATION LINUX SUR ORACLE VM.

## PRÉREQUIS :

Téléchargé et installé le « ORACLE VM VIRTUALBOX »

Lien - <https://www.virtualbox.org/wiki/Downloads>

Téléchargé l’IMAGE ISO Ubuntu 20.04

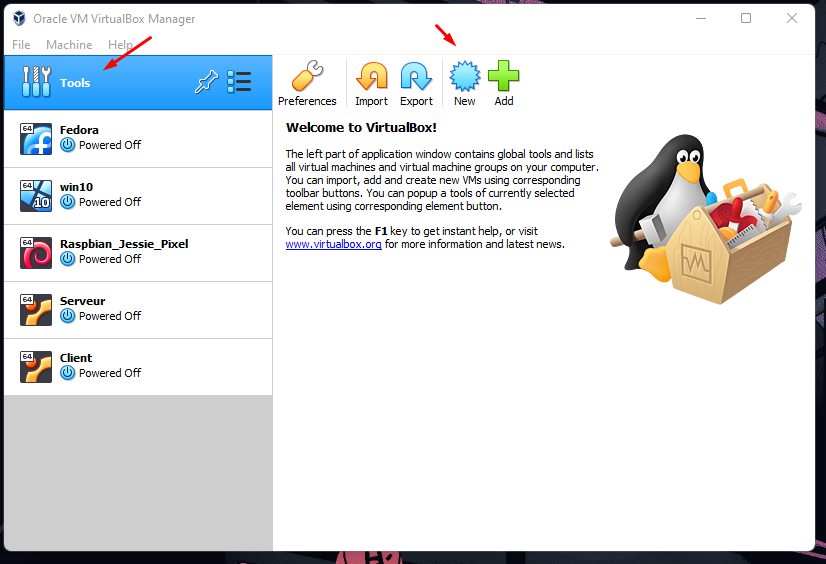
Lien <https://ubuntu.com/download/>

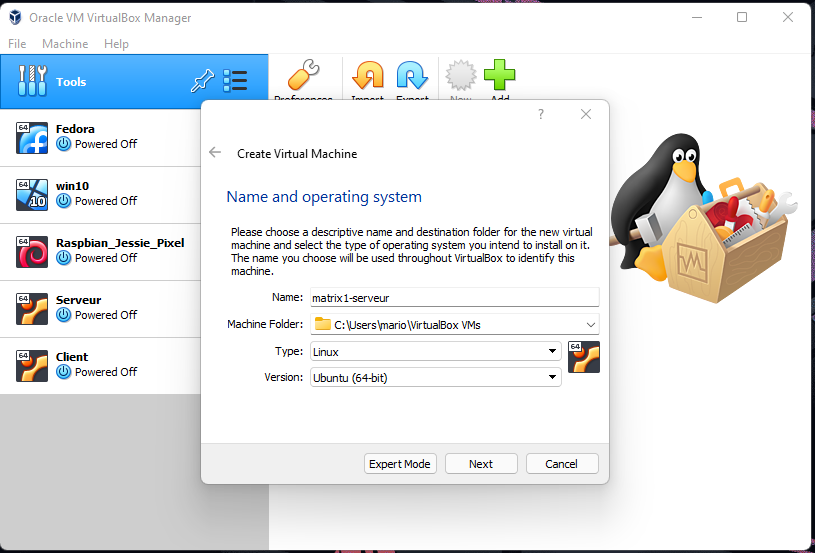
* Serveur Linux – Ubuntu
* Nom : matrix1-serveur
* Matériel : 3Gb RAM ; Disque 20GB
* Client Linux – Ubuntu
* Nom : matrix1-client
* Matériel : 1Gb RAM ; Disque 10GB

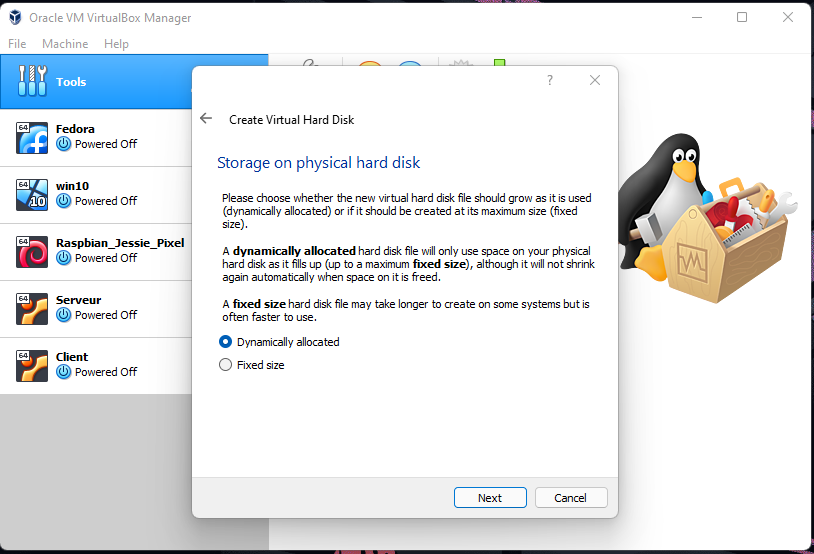
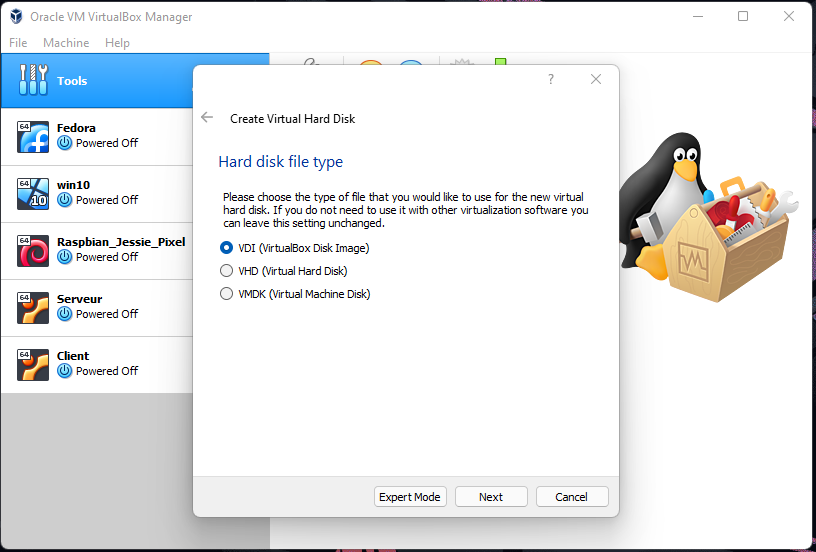
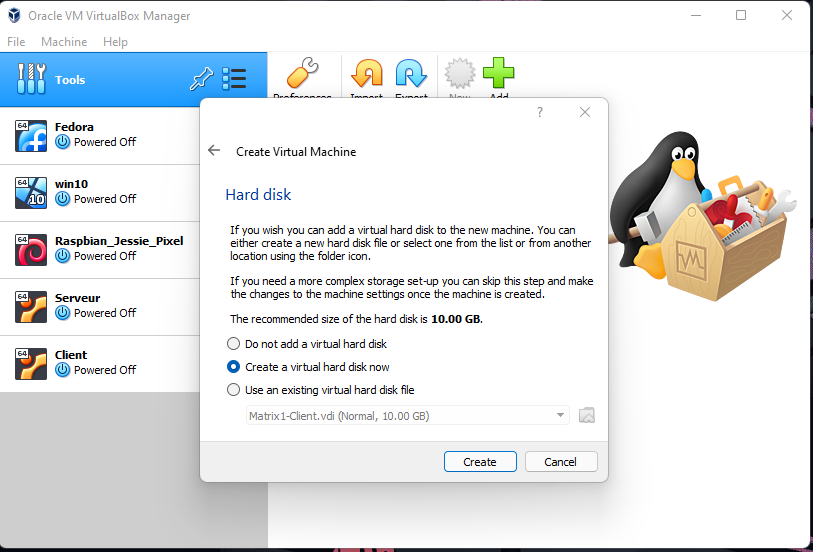
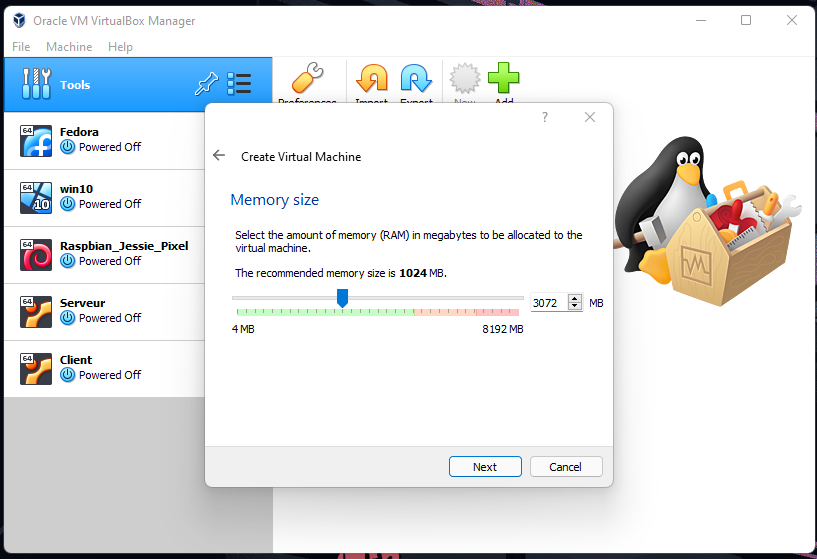
### SERVEUR :

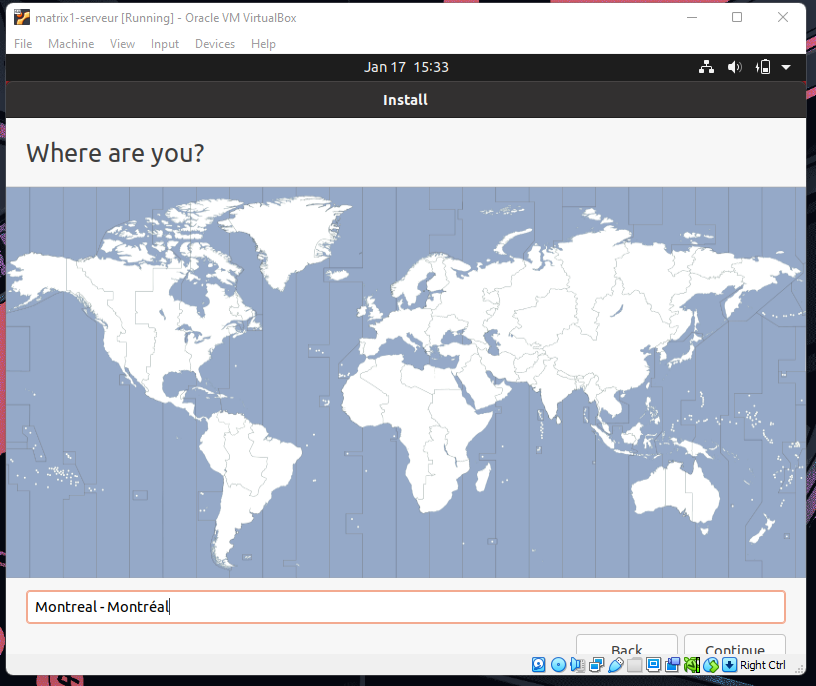
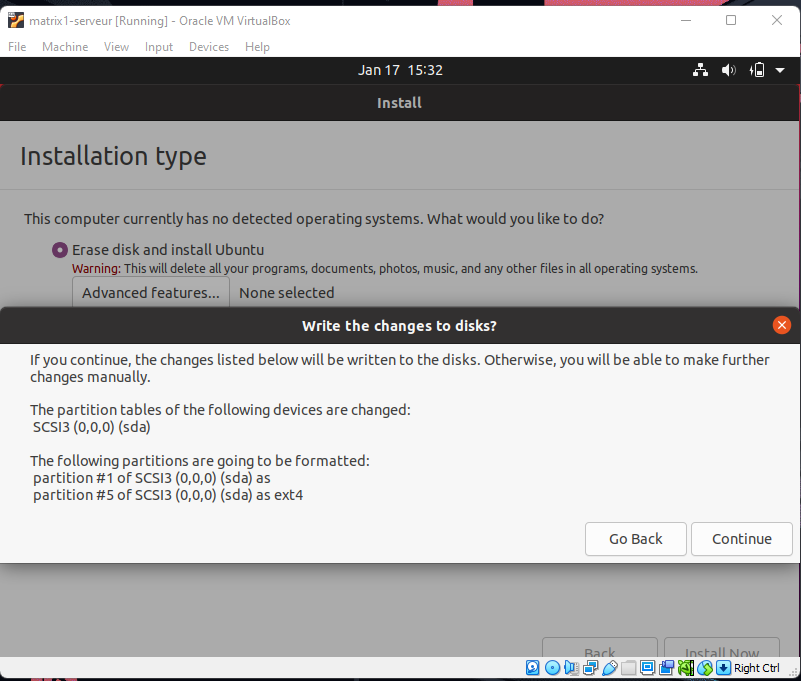
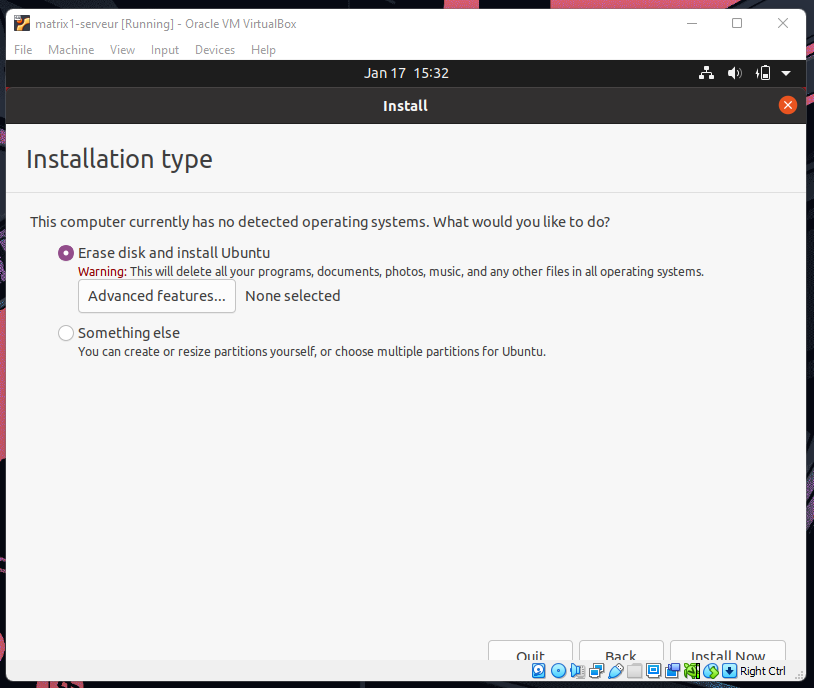
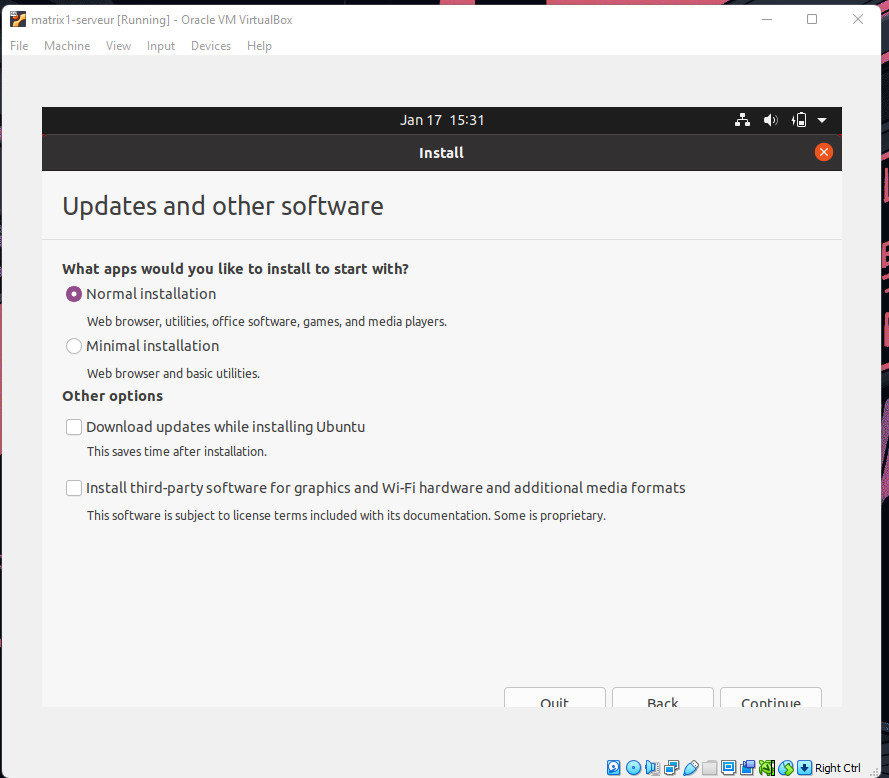
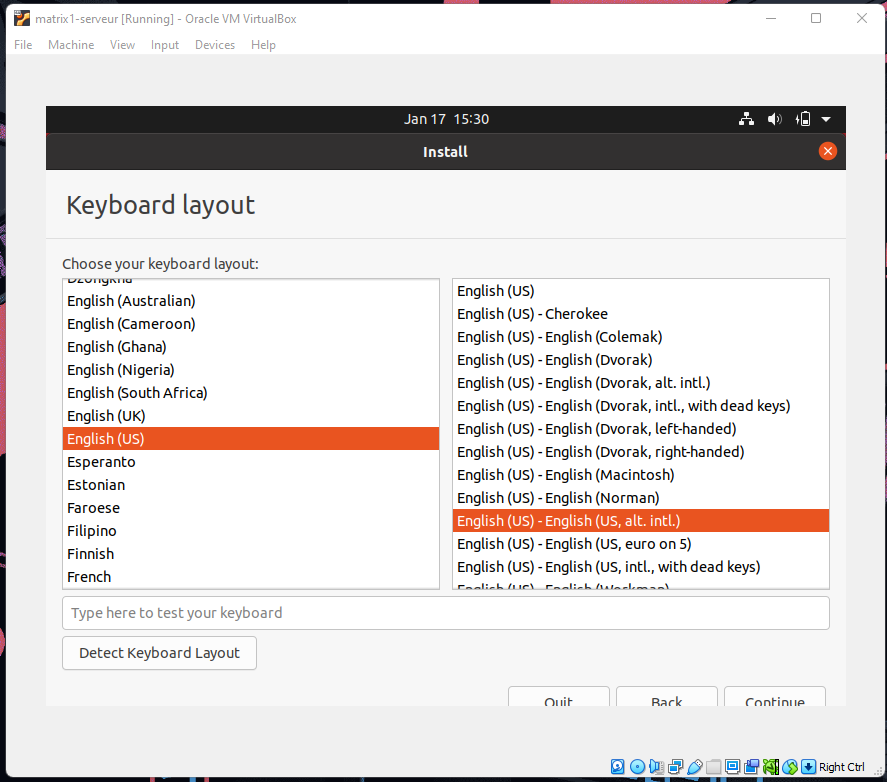
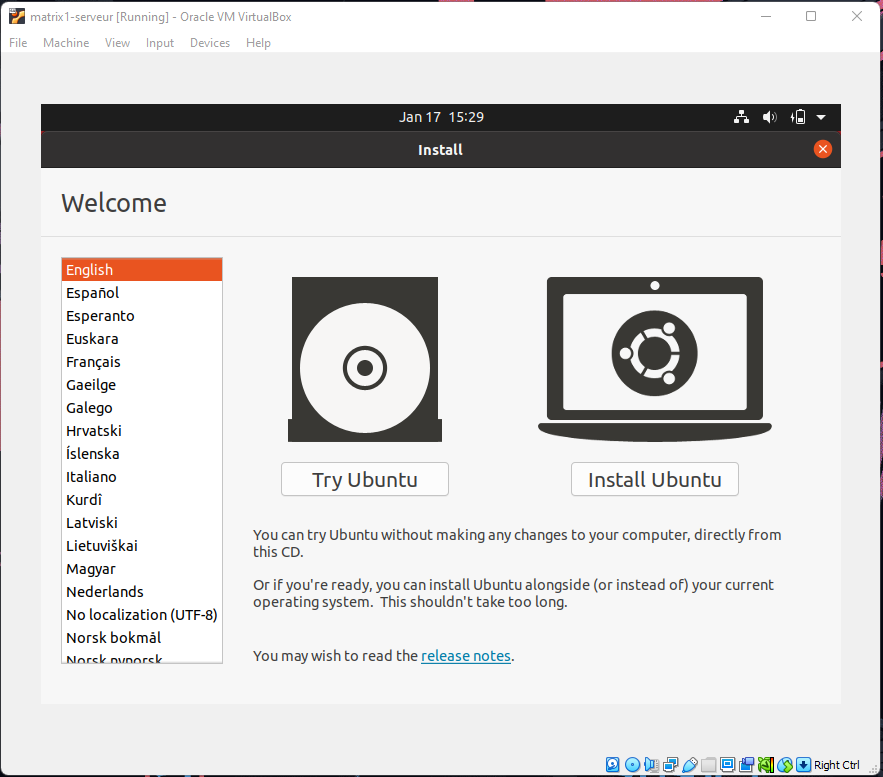
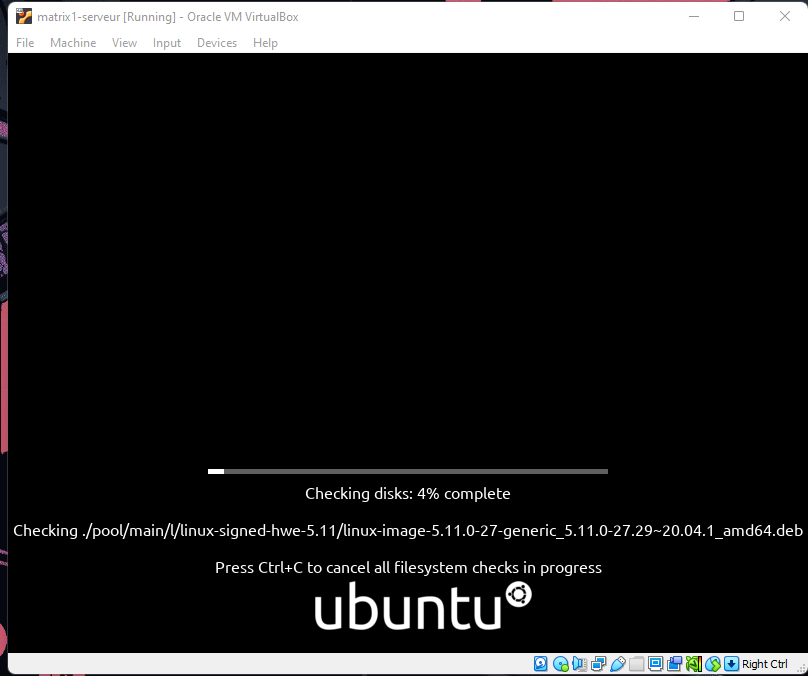
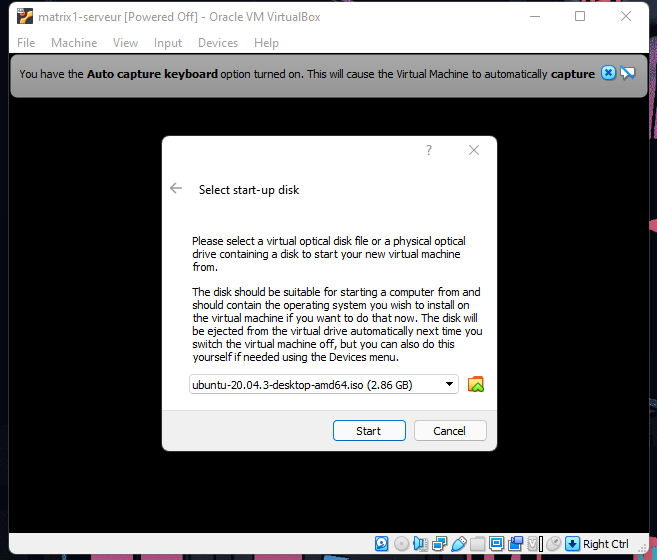
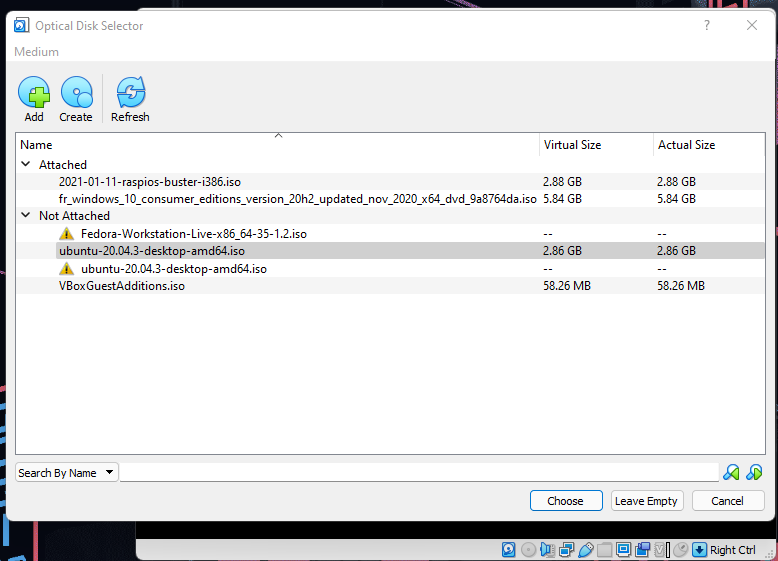
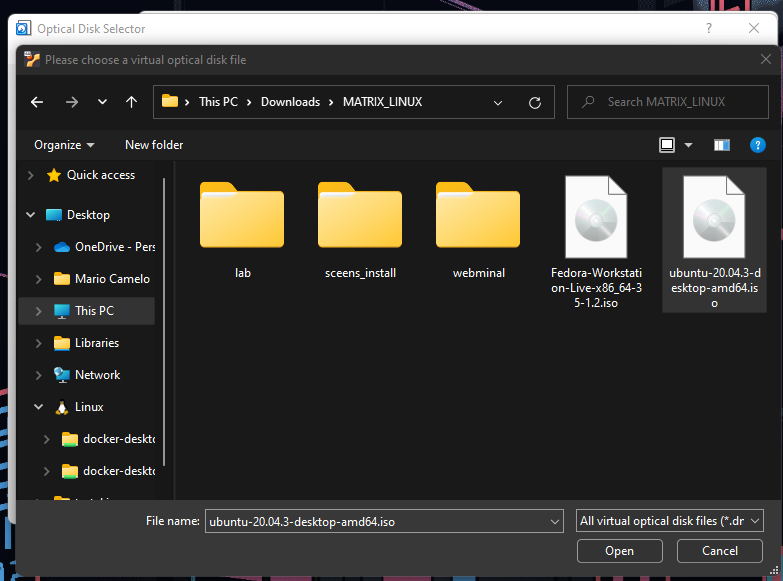
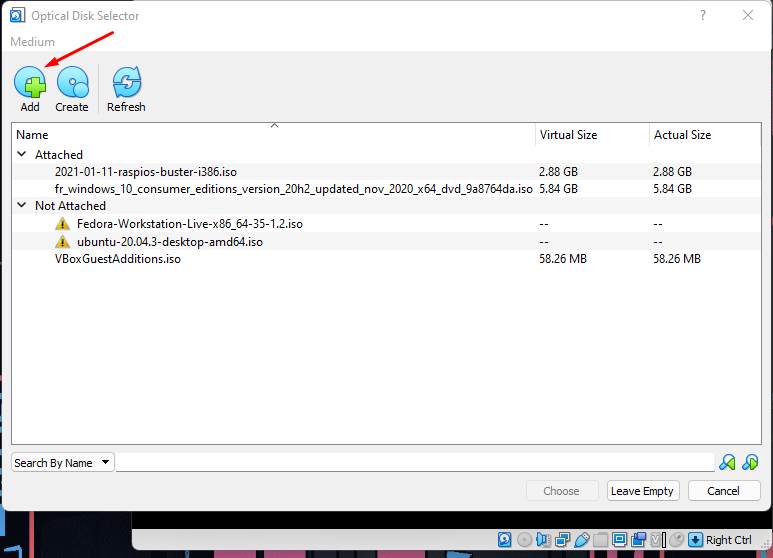
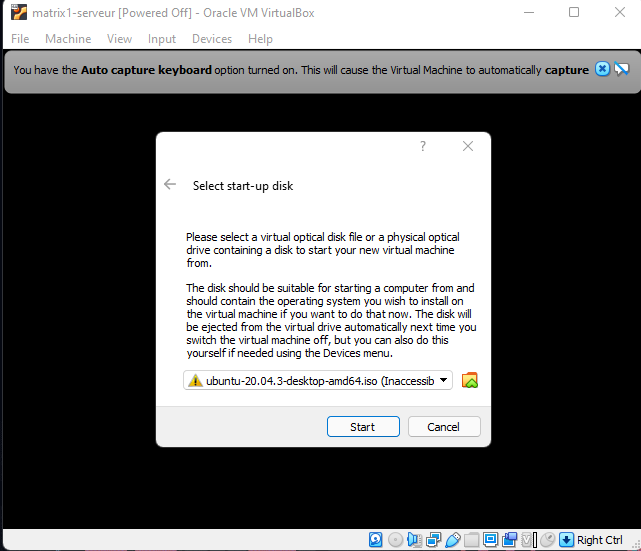
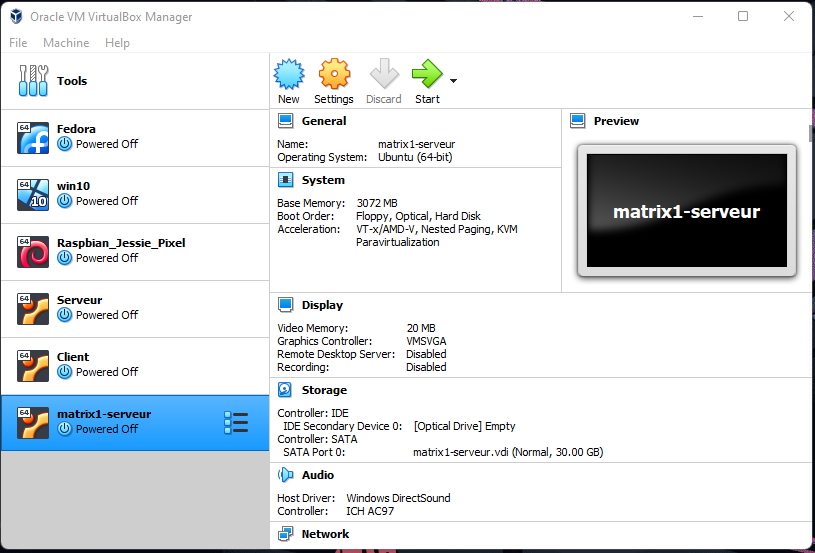
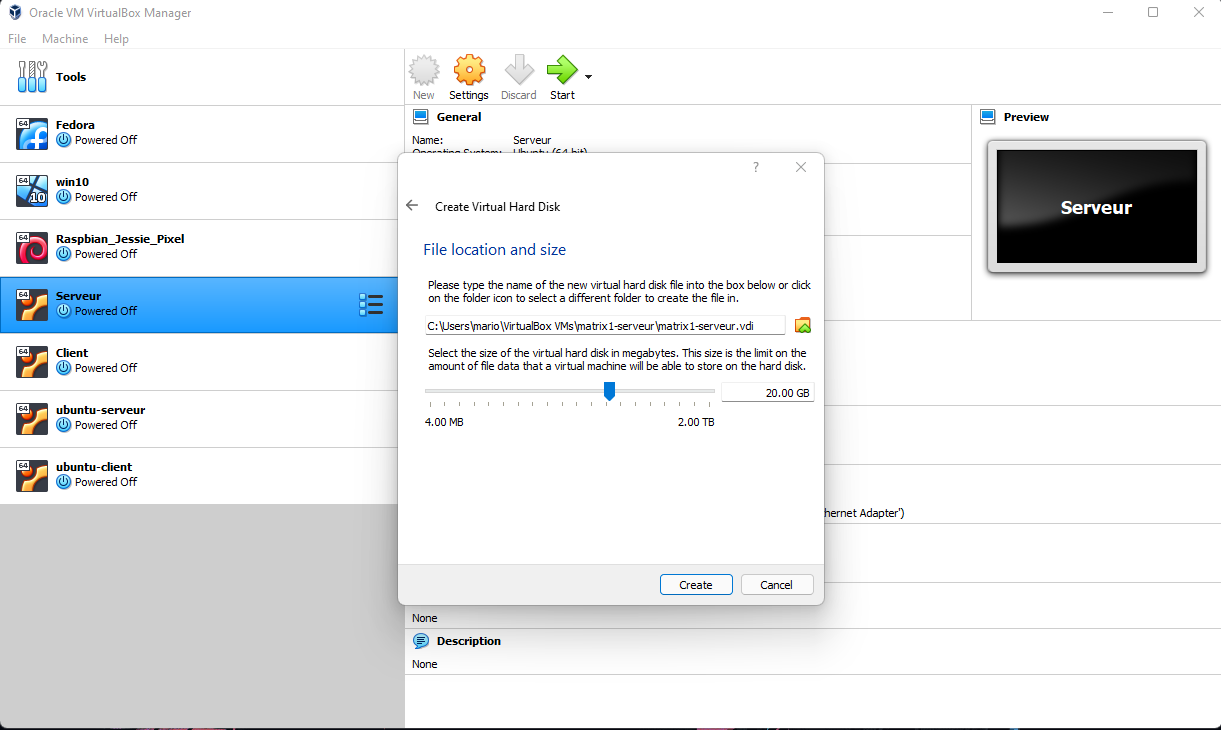
Ici nous allons procedé à la creation du « serveur », choissiez les proprietés qui vous on été demandé, tel que le nom de la machine, mémoire ram, mémoire du disque, le nom de l’utilisateur.

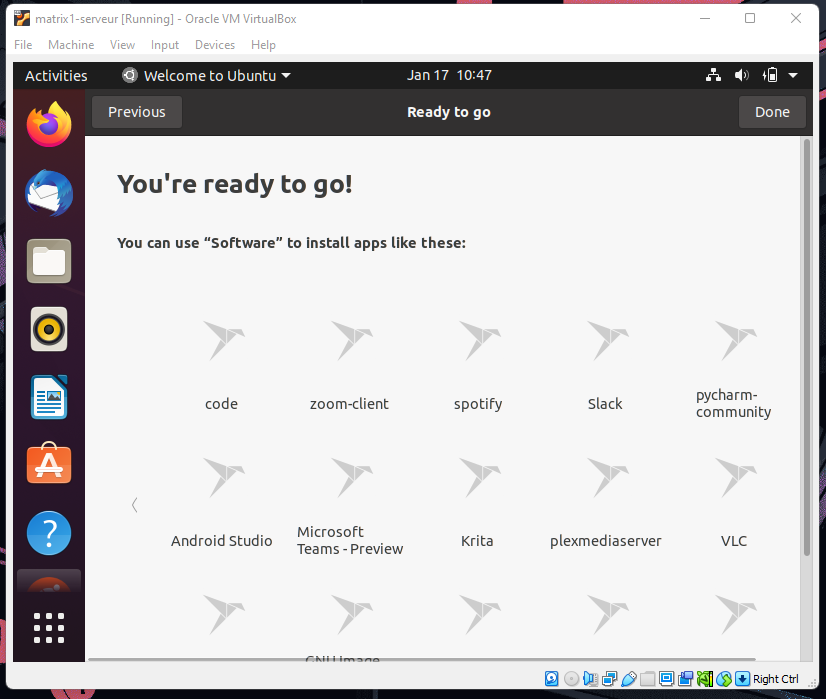
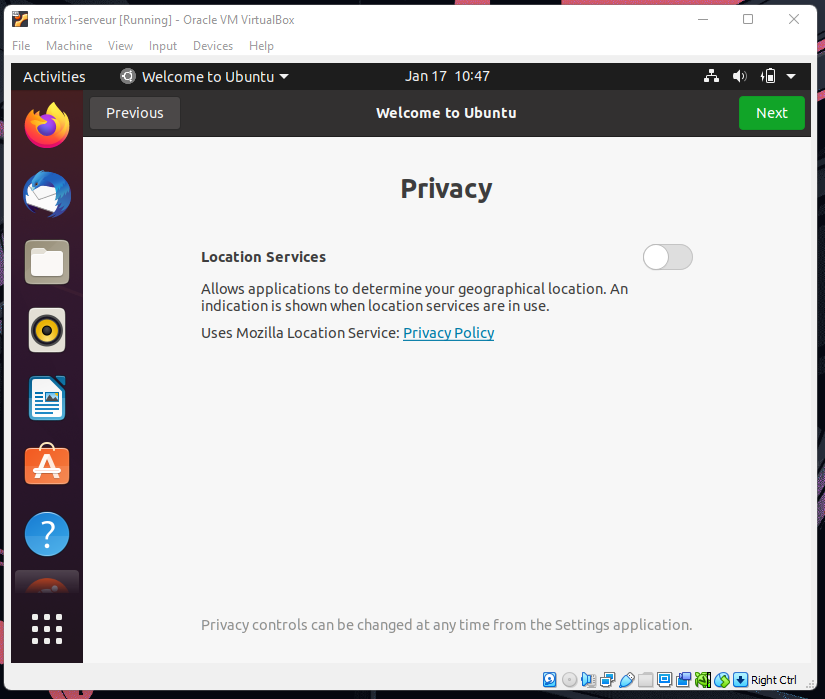
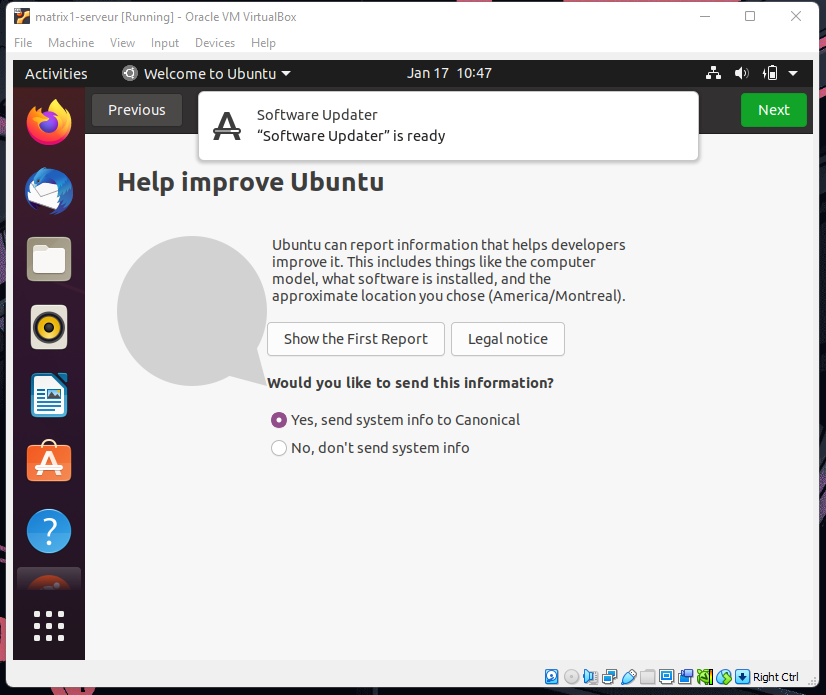
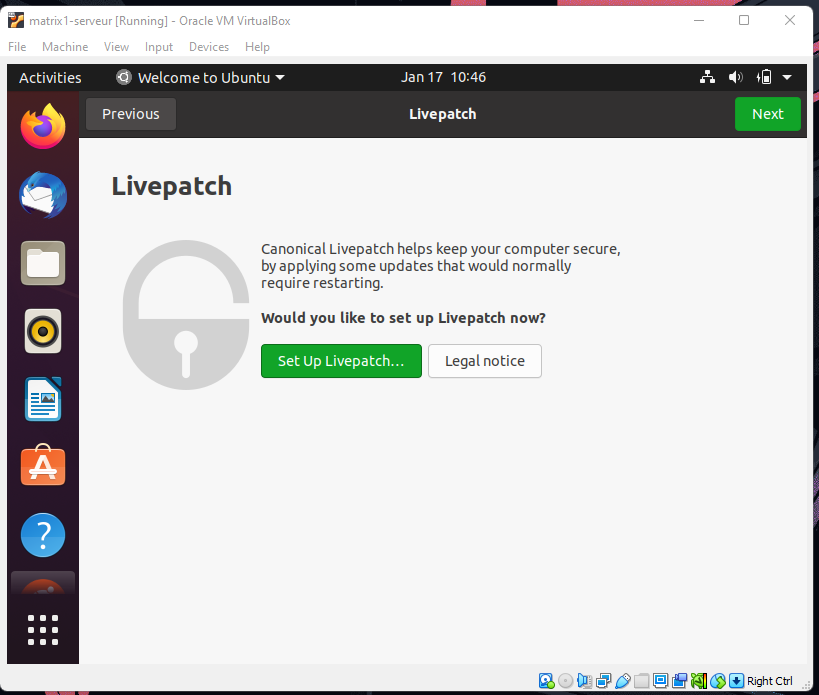
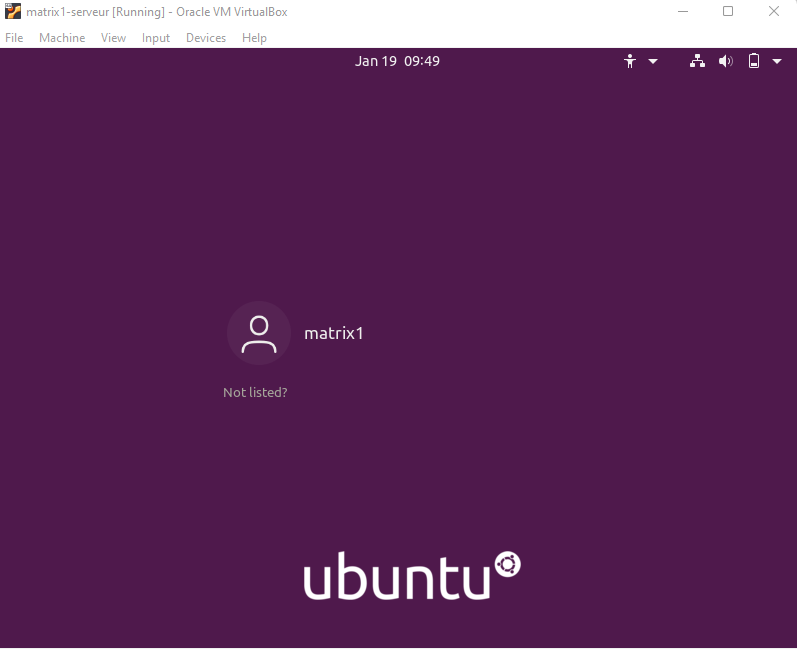
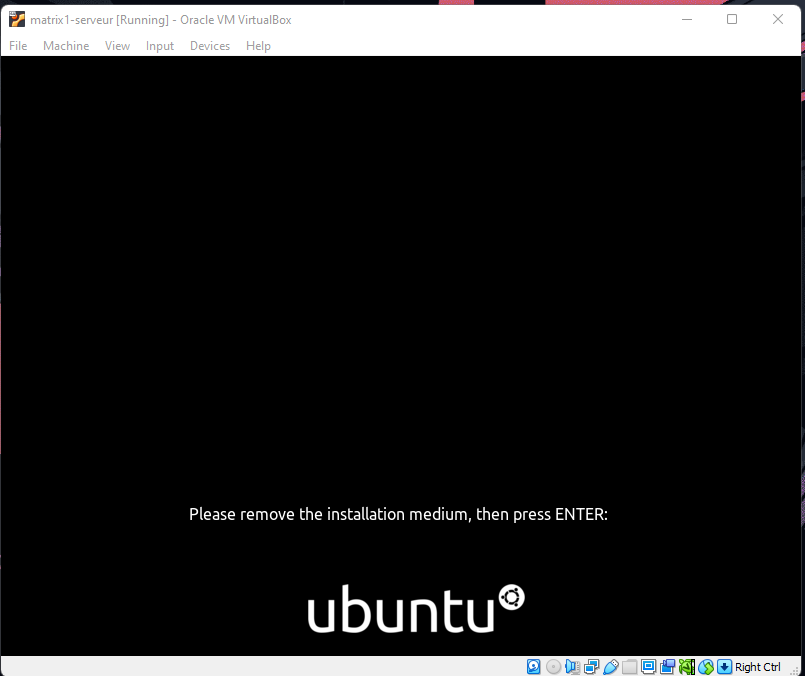
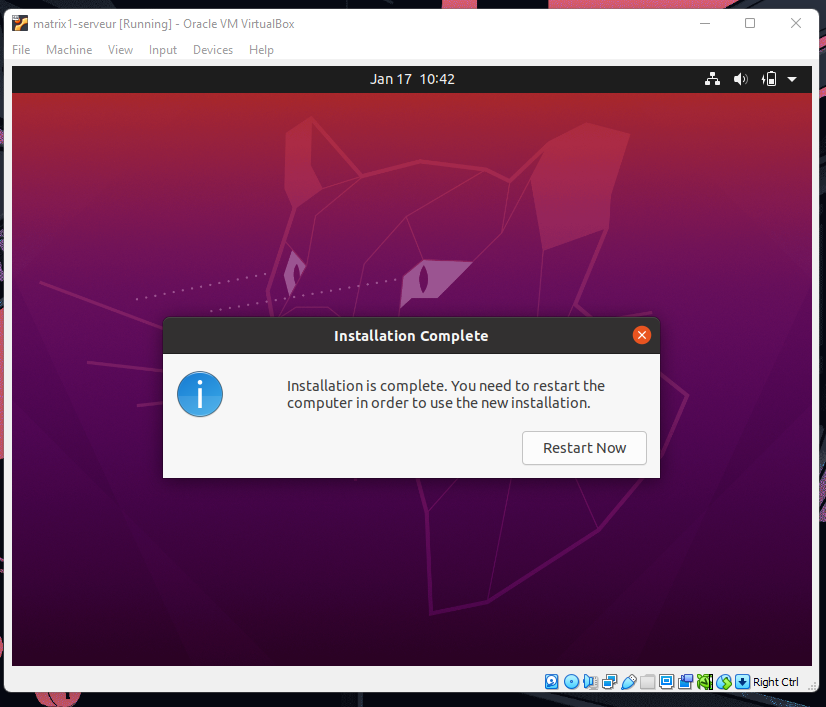
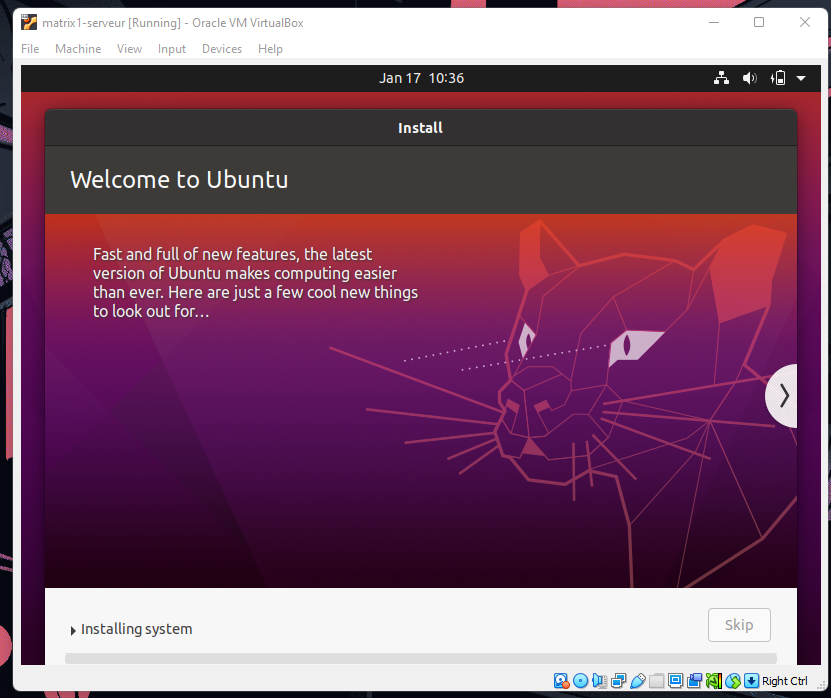
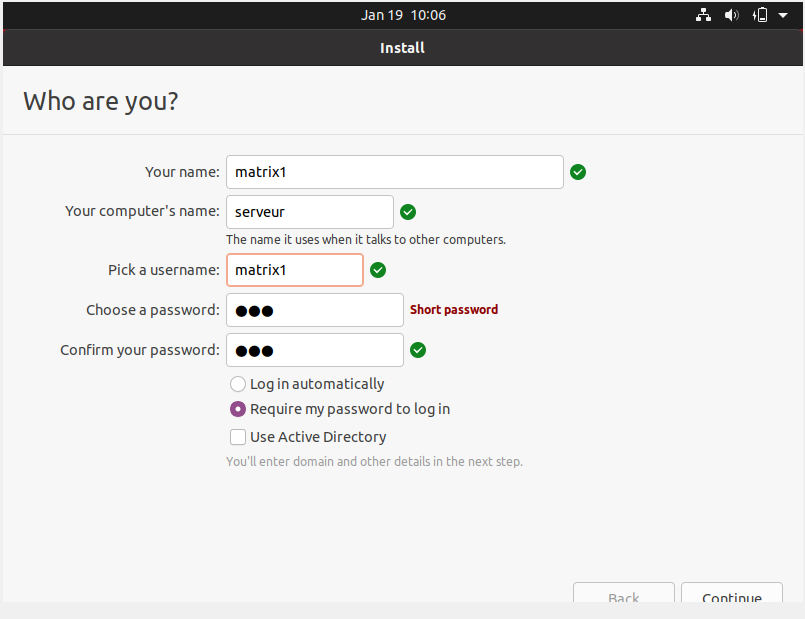
* Serveur Linux – Ubuntu
* Nom : matrix1-serveur
* Matériel : 3Gb RAM ; Disque 20GB





Choisissez 3GB (3072) de mémoire RAM, pour la machine virtuelle « matrix1-serveur ».

Choisissez 20.GB de mémoire DISK, pour la machine virtuelle « matrix1-serveur ».



### 

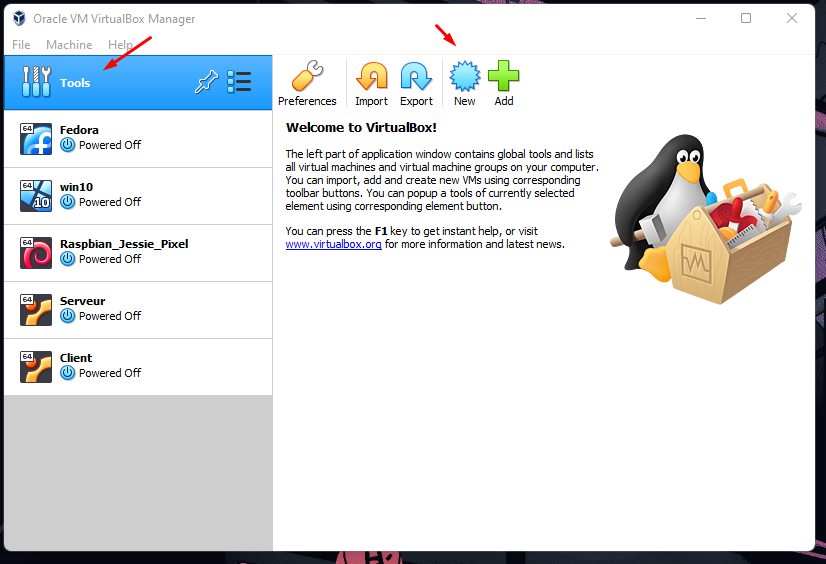
### CLIENT :

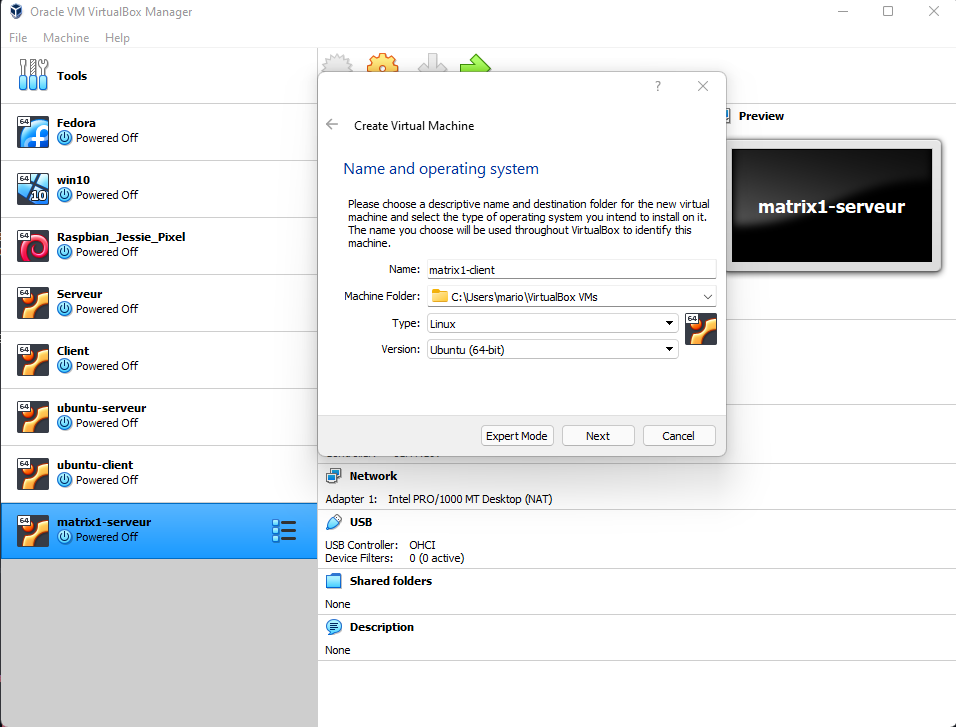
Ici nous allons faire le meme procedé que la machine « serveur » mais en choisissant les propriétés du « client » qui nous on été demandé, comme precedement, le nom de la machine, la mémoire ram, mémoire du disque et enfin le nom de l’utilisateur.

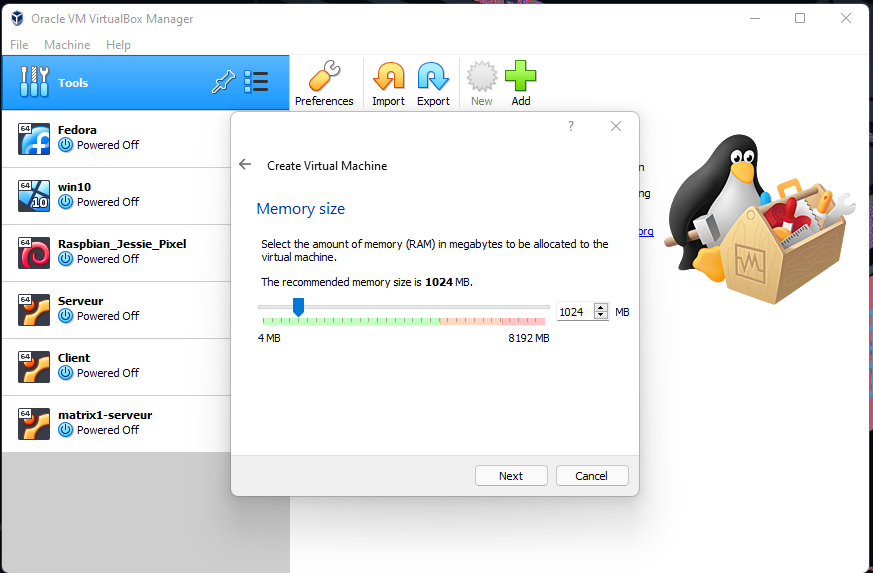
Client Linux – Ubuntu

Nom : matrix1-client

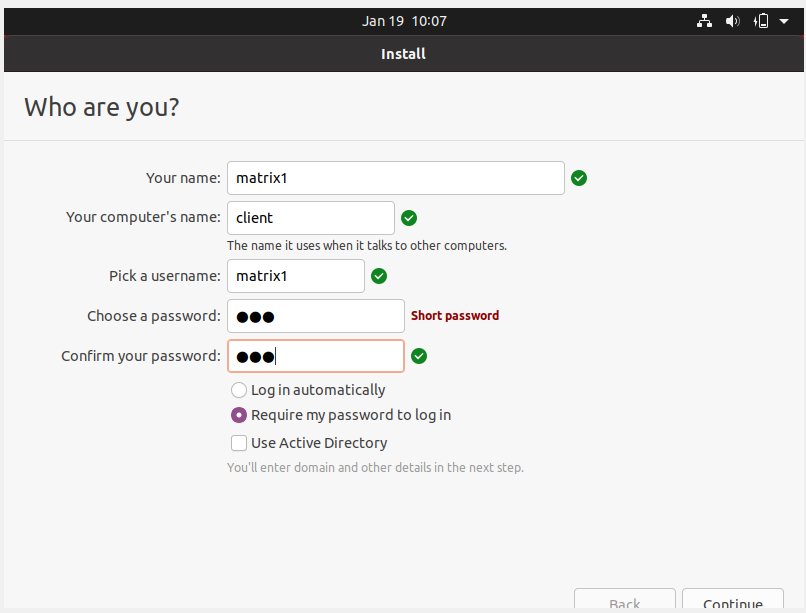
Matériel : 1Gb RAM ; Disque 10GB



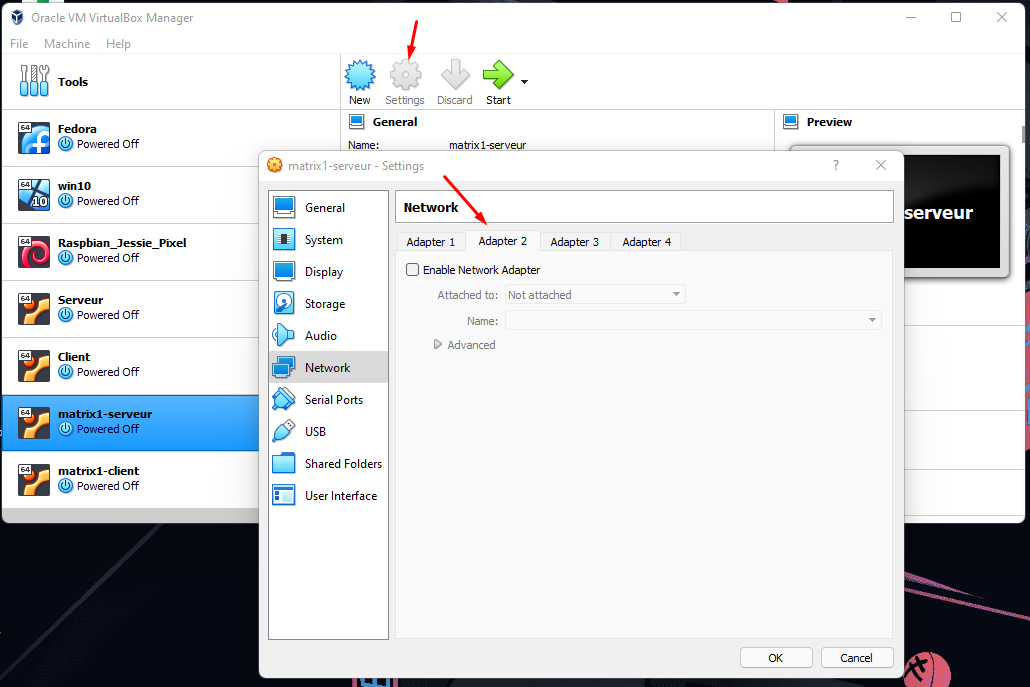
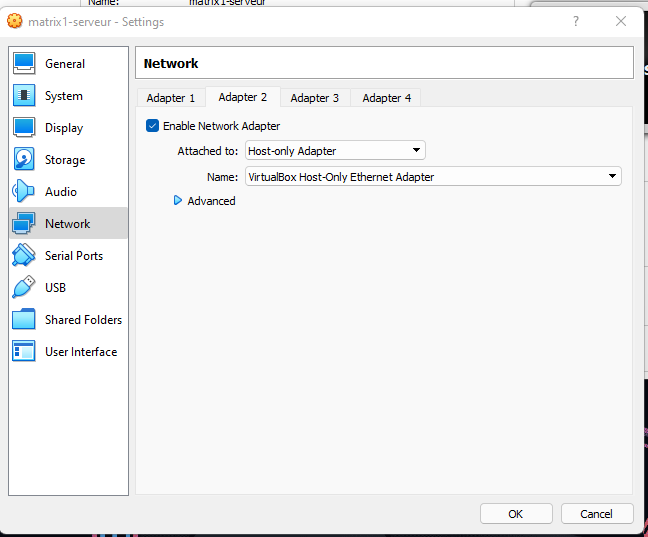


Choisissez 1GB de mémoire RAM, pour le virtual machine matrix1-client.

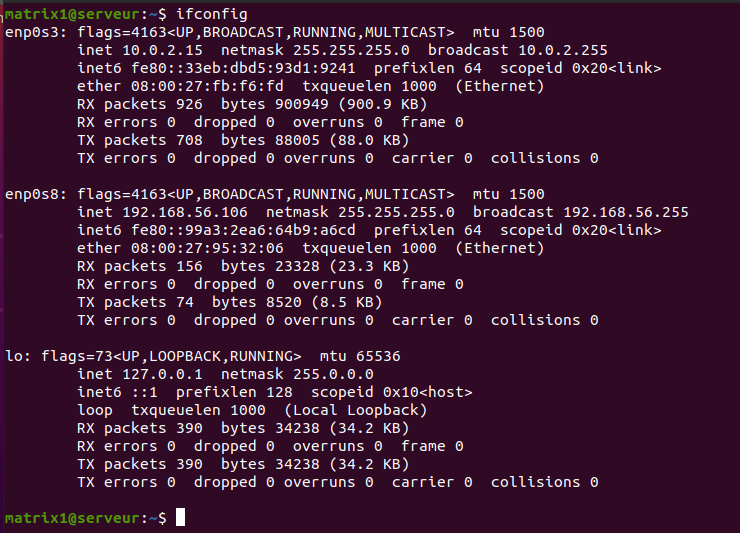
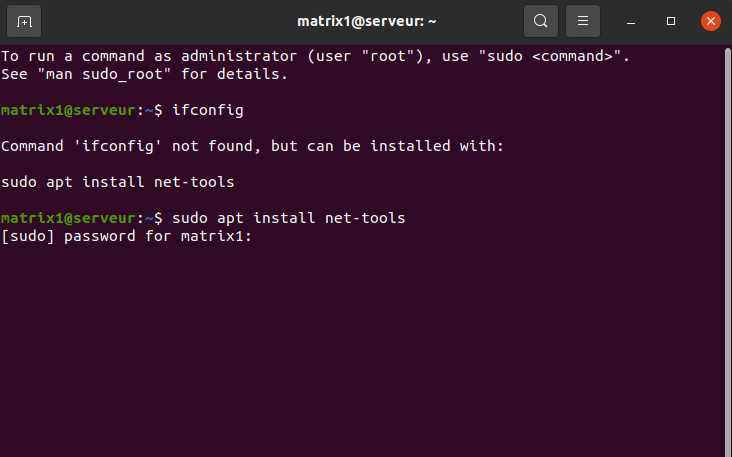
Choisissez 10GB de mémoire DISK, pour le virtual machine matrix1-client.

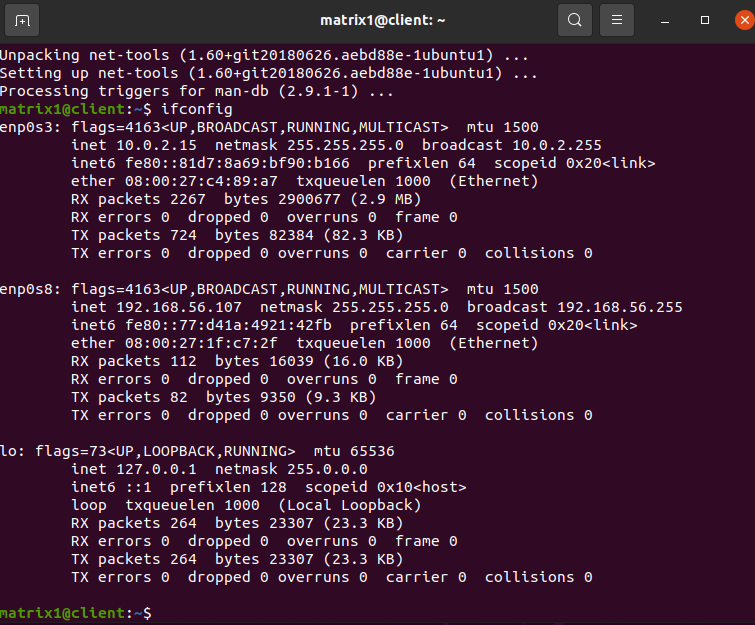


# **COMMUNIQUER LES DEUX MACHINES (SERVEUR ET CLIENT) À L’AIDE DE DEUX ADAPTATEURS RÉSEAU.**

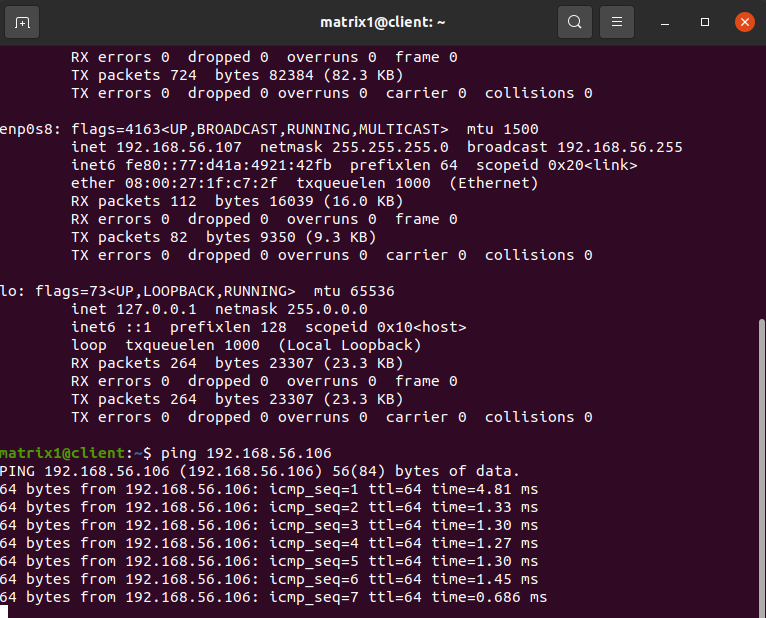
Nous allons ajouter une carte de réseau sur les machines virtuelles afin d’elles puissent communiquer entre elle. Cliquez sur « SETTINGS », ensuite « Network ».  Cliquez sur l’option « ADAPTER 2 » et coché « ENABLE NETWORK ADPTER ». Finalement, choisissez « HOST ONLY ADAPTER ».

Pour valider le bon déroulement de l’installation, nous pouvons utiliser la commande « ifconfig » depuis le TERMINAL SHELL.





Pour valider la communication entre les deux machines virtuelles, on peut utiliser le command : « ping <IP> ».



Nous allons créer les variables de hosts afin de rendre plus simple l’utilisation des « IP adress ».

#CLIENT

sudo pico /etc/hosts

#add <ip-server> <nickName>

192.168.56.106 server serveur

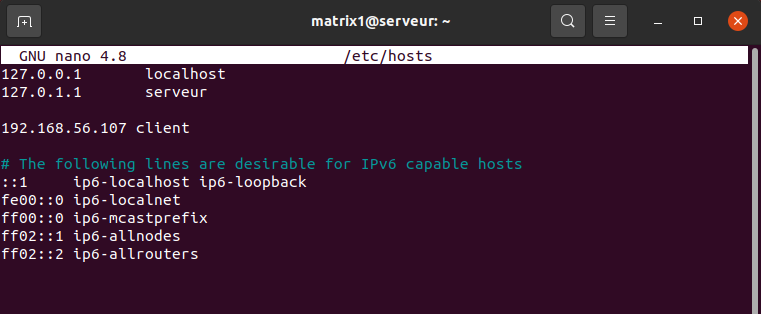


#SERVEUR

sudo pico /etc/hosts

#add <ip-client> <nickName>

192.168.56.105 client



Confirmer avec la commande: ping <nickName>

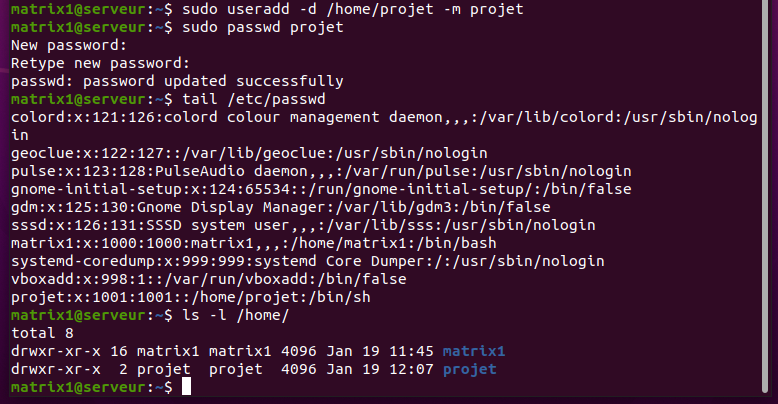
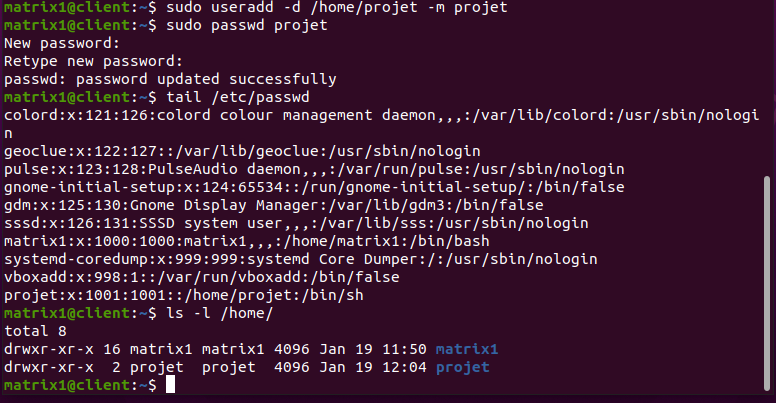
Ex:

ping client

ping server

# **CRÉER L’UTILISATEUR « PROJET ».**

Nous allons ajouter un nouvel utilisateur avec la commande « useradd » et créer son mot de passe avec « passwd ».



# **GÉNÉRER UNE CLÉ SSH SUR LE SERVEUR POUR L’UTILISATEUR DU PROJET.**

Prérequis:

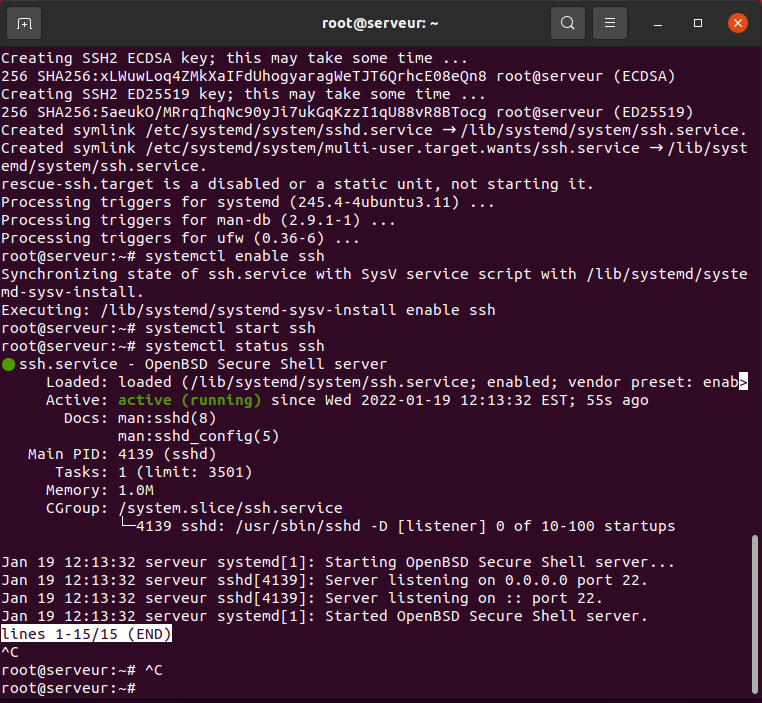
sudo su –

apt install openssh-server

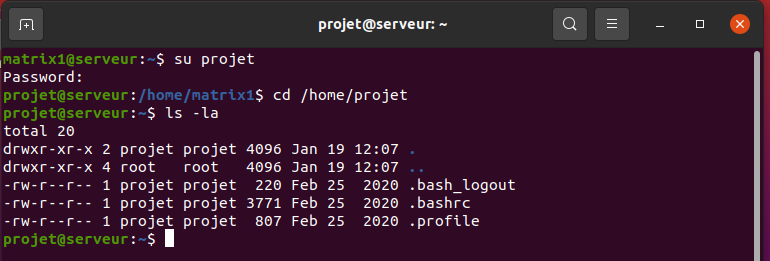
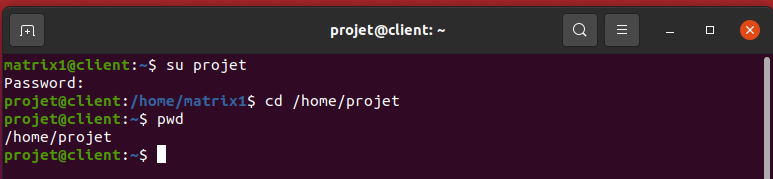
systemctl enable ssh

systemctl start ssh

systemctl status ssh

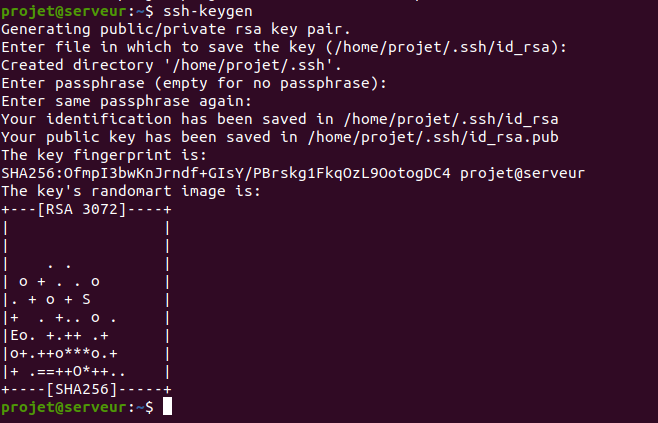


Utilisez la commande « exit », pour sortir de « root ». Après roulez la commande « su projet » pour utiliser l’user : projet.



ssh-keygen

#N’entrez pas de mot de passe.



# **Copier la clé publique sur le client.**

Après la création de la clé publique, du côté du client, nous allons la copier vers le serveur. Après la transmission de la clé sur le client, on peut voir le fichier, « authorized\_keys »

#CLIENT

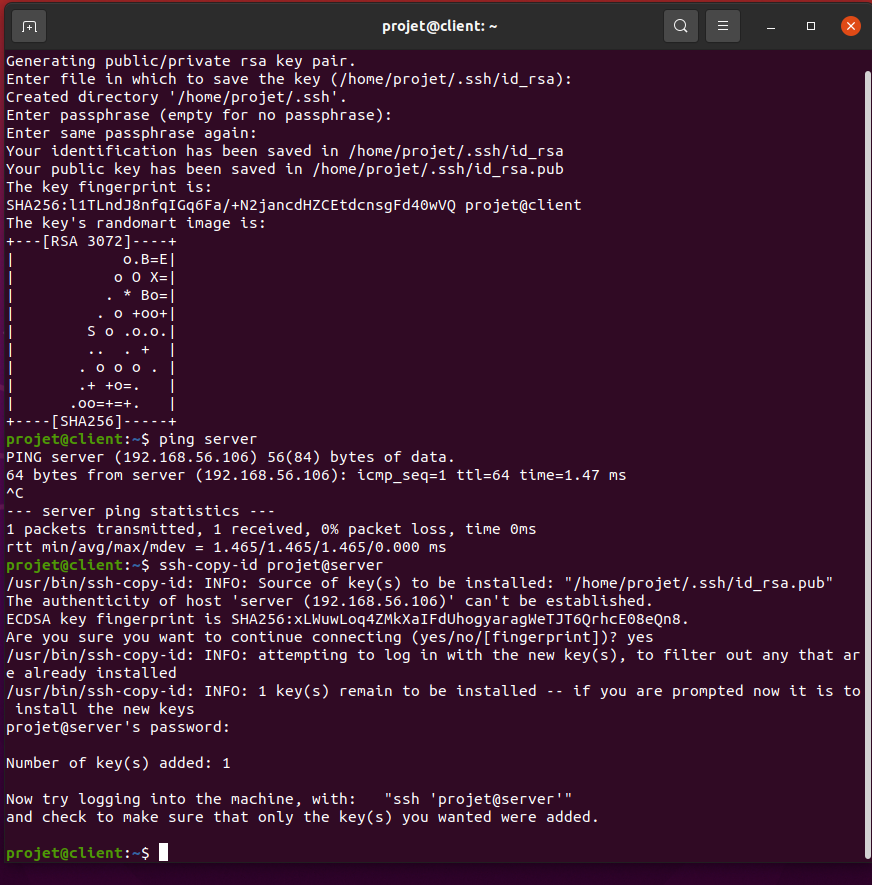
#ssh-copy-id <user-destination>@<ip-destination>

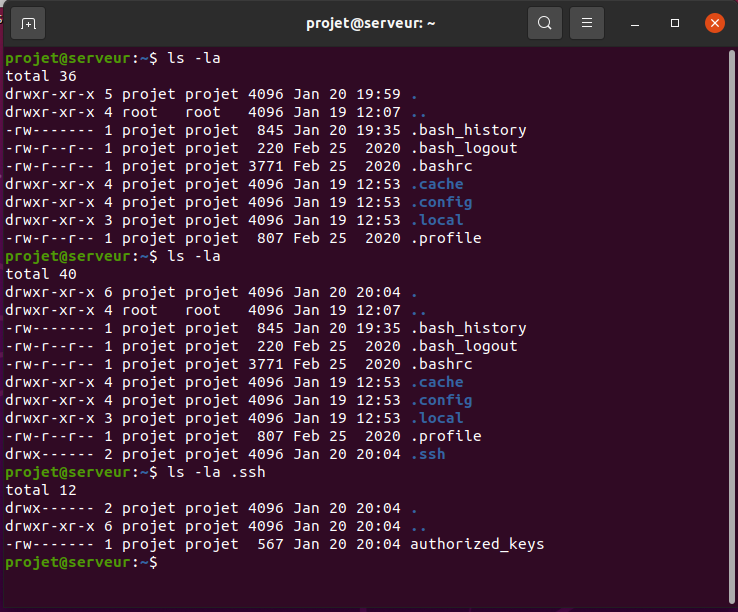
ssh-keygen

ssh-copy-id projet@server

# On peut utiliser le nom client, parce qu’on a fait l’édition sur /etc/hosts , ou

ssh-copy-id [**projet@192.168.56.10**](mailto:projet@192.168.56.107)**6**





# **Valider qu`il y a bien communication entre le serveur et le client en transmettant le fichier /home/projet/test.txt du client au serveur.**

Prérequis:

###SERVEUR

#Sortir de l’utilisateur « projet », avec la commande « exit ».

sudo apt install vsftpd -y

systemctl start vsftpd

#Logger vous avec « projet » : su projet | cd /home/projet

###CLIENT

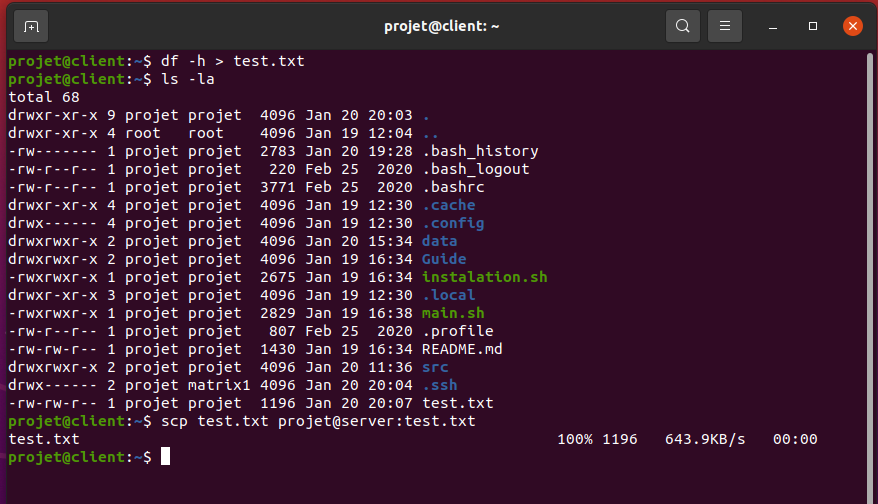
#Creer un fichier.

df -h > test.txt

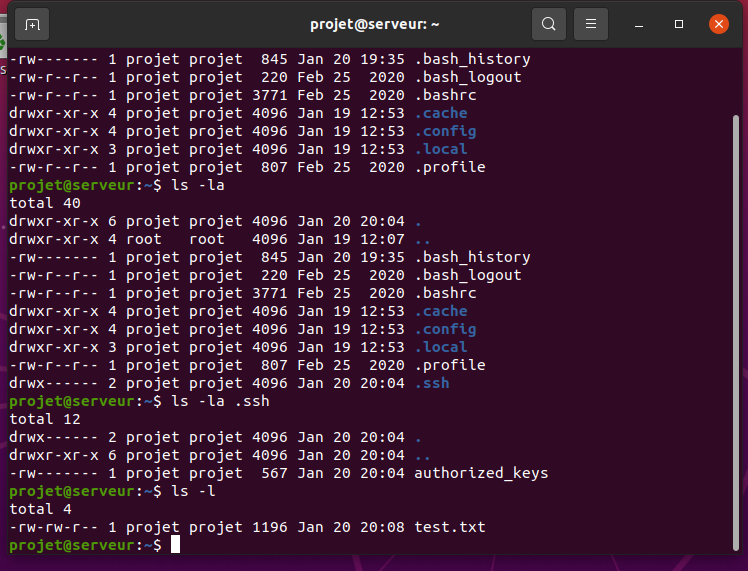
#Envoyez le fichier avec la commande « scp ».

#Ex: scp <nomFichier> <nomUtilisateur>@<ipServer> :<nouveuNomeFichier>

scp test.txt [projet@server:test.txt](mailto:projet@server:test.txt)



Nous pouvons voir et validé le fichier qui été transféré.

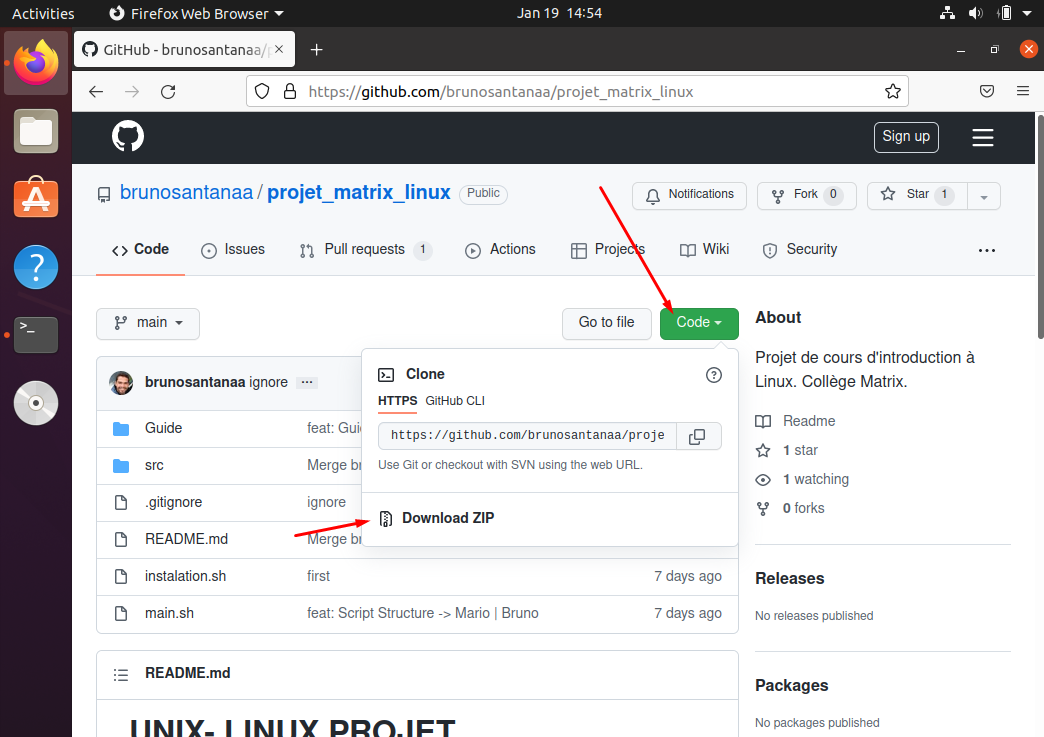


# **LE SCRIPT.**

## PRÉREQUIS:

Télécharger le projet compacté.

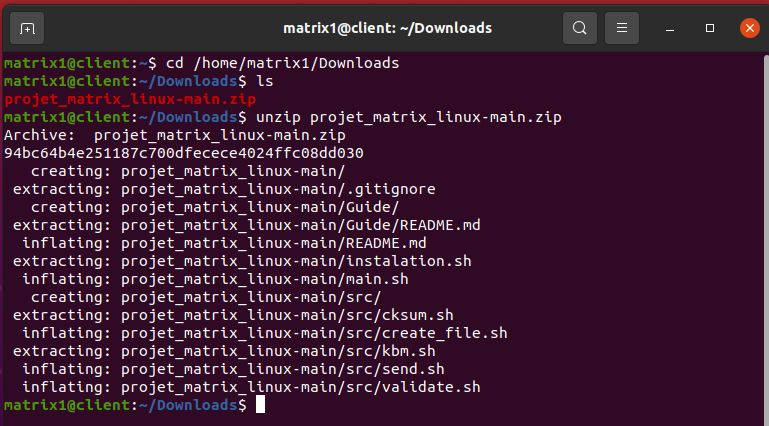
Lien : <https://github.com/brunosantanaa/projet_matrix_linux>



Sur le site, vous allez cliquer sur le Button « CODE », et ensuite sur « Download ZIP » pour lancer le téléchargement.

Dans le Terminal, allez au PATH de Downloads.

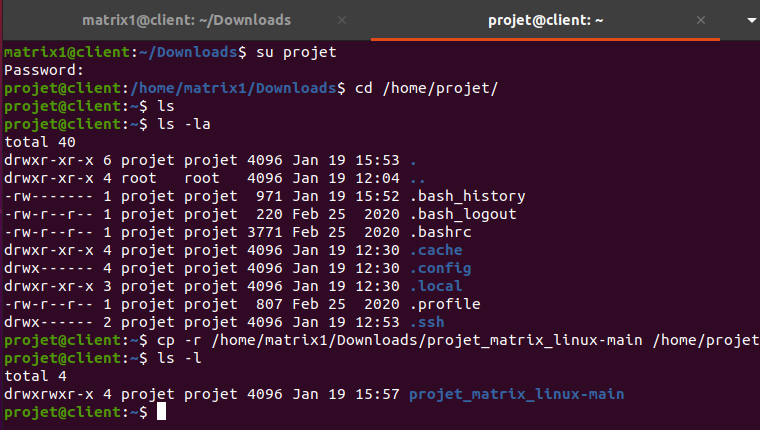
Utilisez la commande : cd /home/matrix1/Downloads



Maintenant nous copierons le répertoire de l’utilisateur « matrix1 » à « projet ».

Alors vous pourrez vous logger avec « projet » et allez au chemin du projet. /home/projet/projet\_matrix\_linux-main/

cp -r projet\_matrix\_linux-main /home/projet



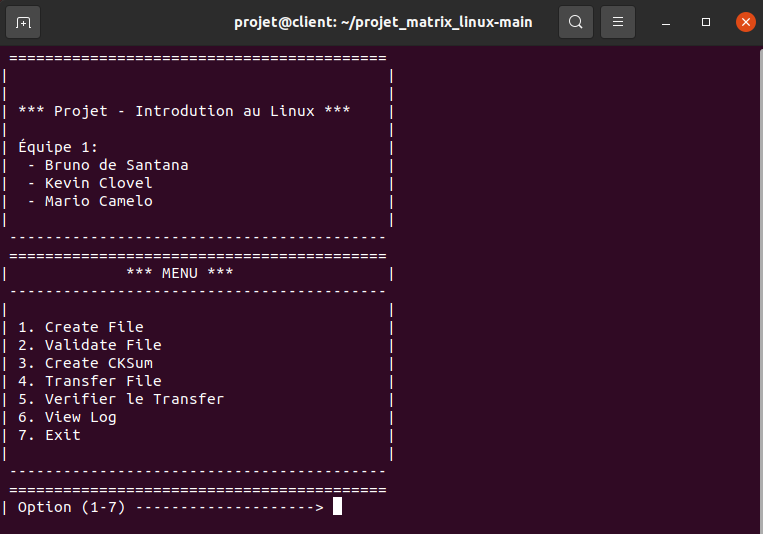
mv projet\_matrix\_linux-main/\* /home/projet

Le but de se script est d’avoir un MENU, et des commandes automatisées pour exécuter les tâches suivantes.

## MENU SUIVANT SOUS LINUX.

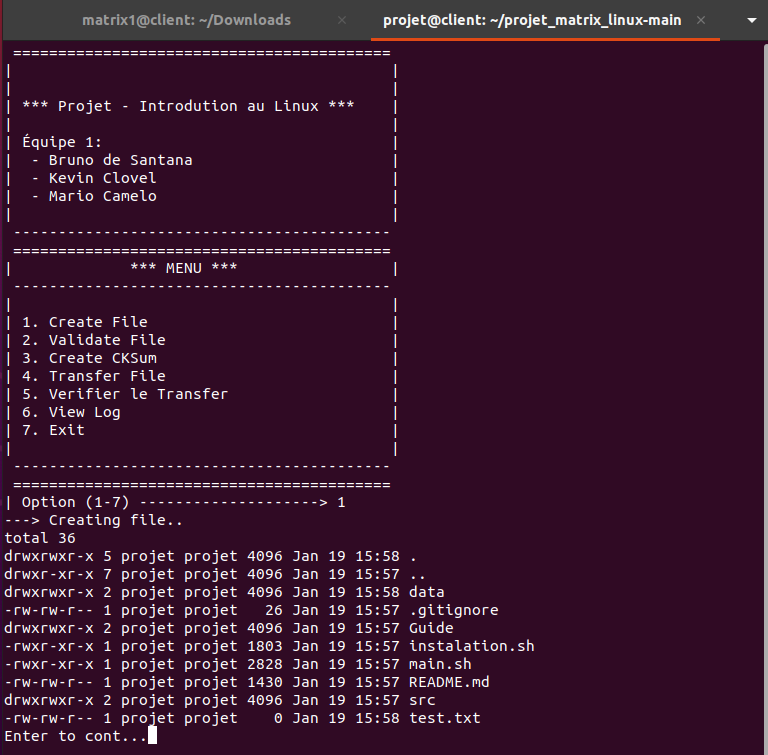
* Générer le fichier à copier.
* Valider le type de fichier généré.
* Générer le cksum à partir du fichier. (Nombre qui résume le contenu du fichier de manière unique)
* Transmettre le fichier du client au serveur.
* \*Valider le cksum à partir du fichier du serveur.
* Journal de transmission.

Exécuter le script avec la commande : ./main.sh

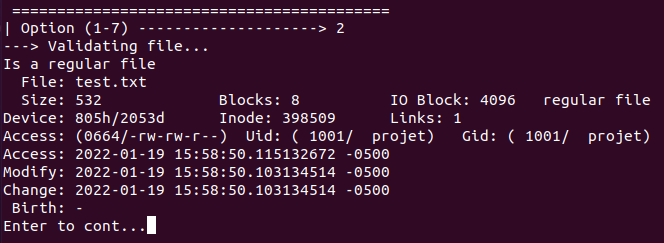


Dans le MENU, vous aurez 7 options dont il vous faudra en suivre l’ordre séquentiel de 1 à 7 afin de réussir le parcheminèrent des commandes.

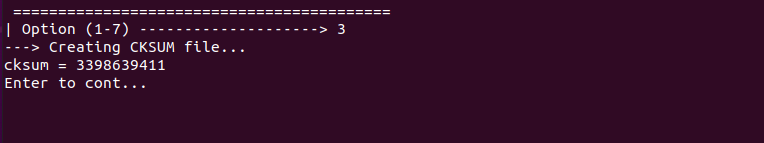
L’option 1 : le script va créer un fichier qui s’appellera : test.txt.



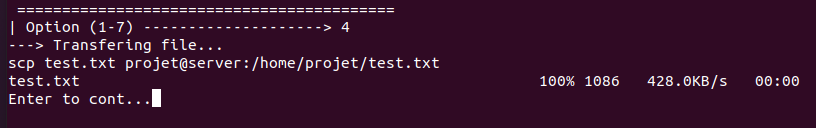
L’option 2 : le script validera le type du fichier test.txt.



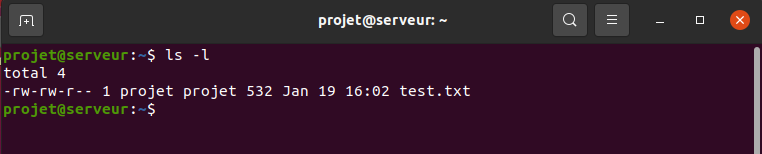
L’option 3 : le script générera le cksum du fichier test.txt.



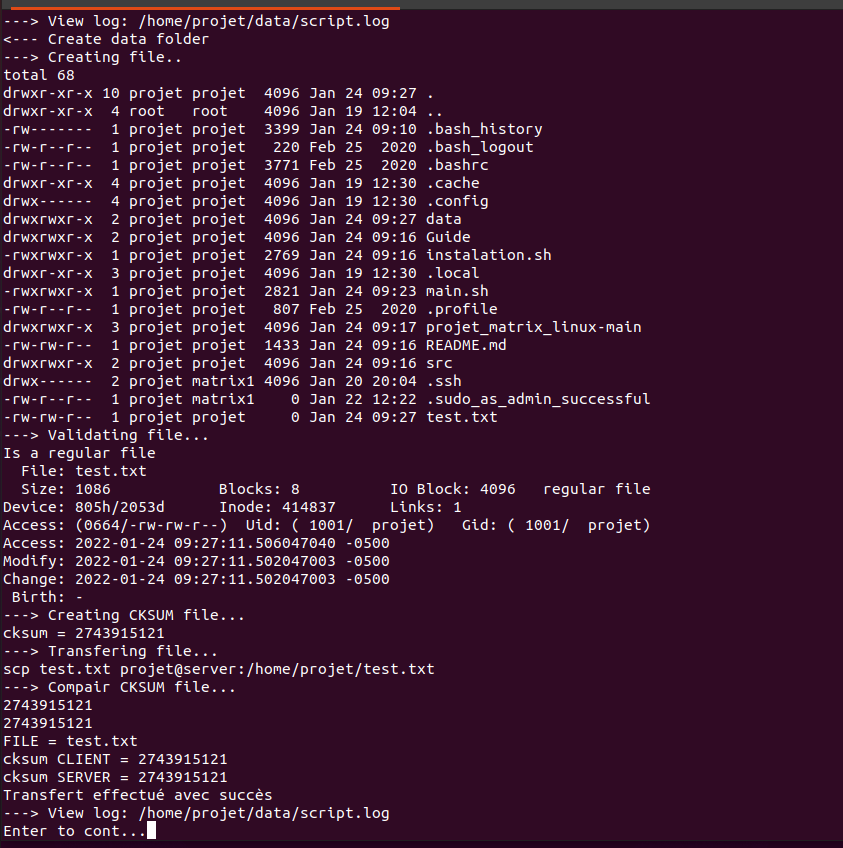
L’option 4 : le script va transmettre le fichier test.txt du client au serveur.



L’option 5 : le script va valider si le fichier test.txt a bien été transmis sans perte de données.



L’option 6 : visualiser le journal.



L’option 7 : Qui comme son nom l’indique, permet de sortir du script.



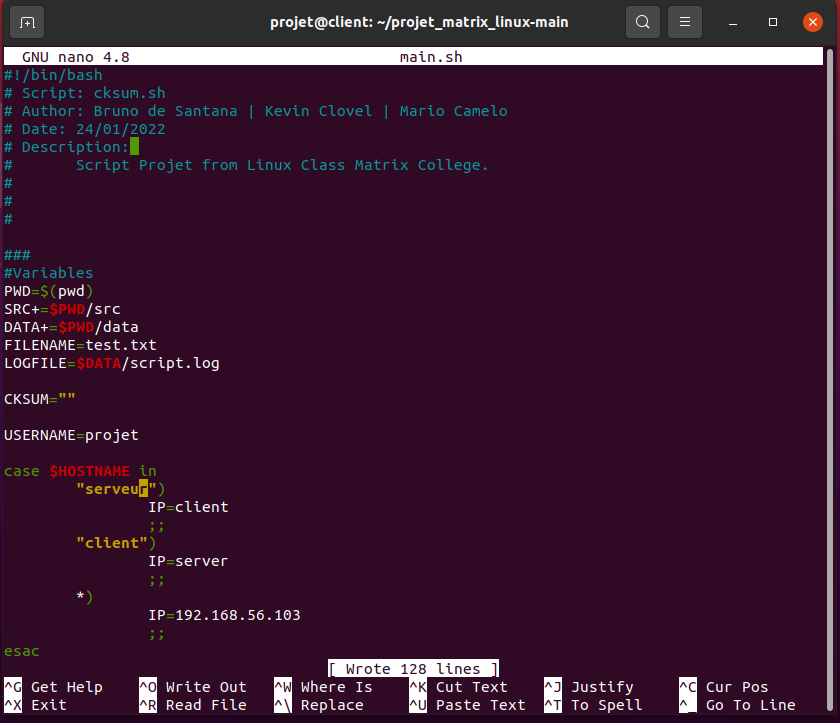
### 

### LA SCRUCTURE DU SCRIPT.



## LEXIQUE DES CODES.

### MAIN.SH



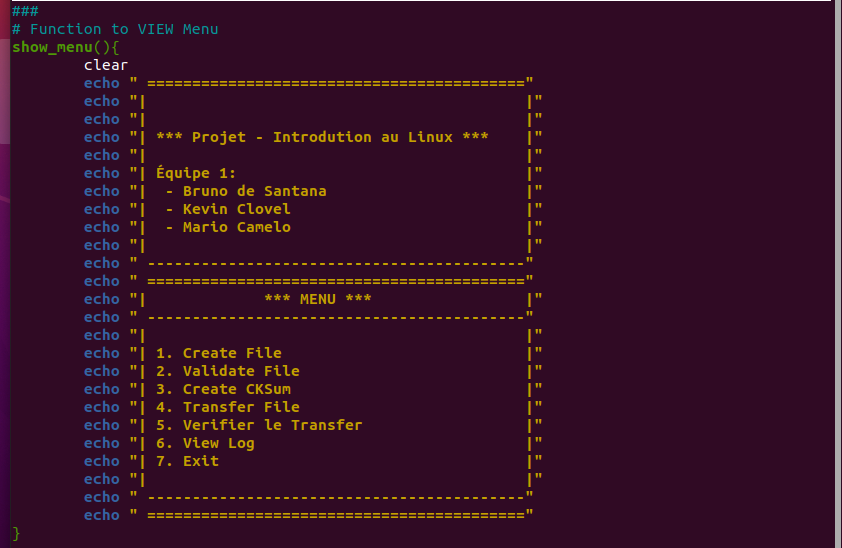
Les variables ont été défini.



Ici, nous utilisons la commande case pour définir la variable IP qui va contenir l’adresse de la machine cible, en fonction de la variable hostname qui contient le nom de la machine hôte. Si la machine hôte est « serveur » alors l’adresse de la machine cible sera « client », que nous avons défini dans /etc/hotsts.

show\_menu() :

Efface l’écran (CLEAR) et afficher sur l’écran tout la ligne avec la commande ECHO.



wait() :

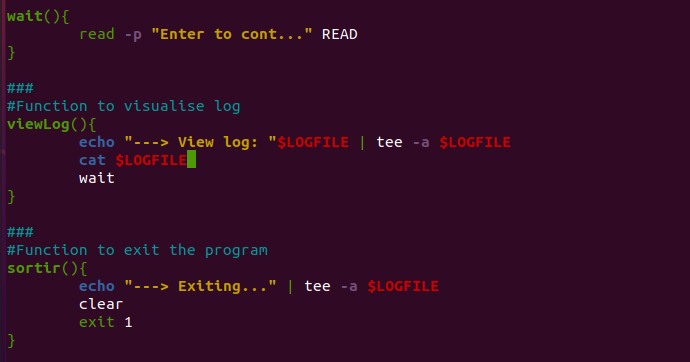
Permettra d’attendre que l’utilisateur appuie sur une touche pour continuer. (READ)

viewLog() :

Affichera le journal d’utilisation du programme. (CAT)

Sortir() :

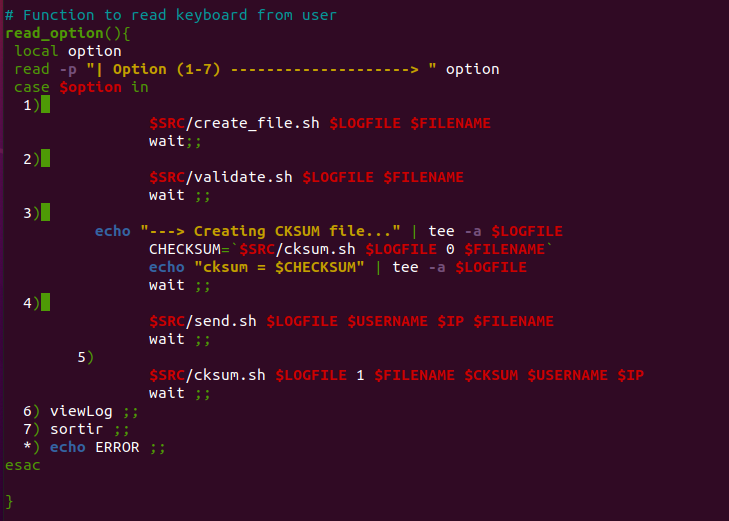
Utilise la commande EXIT, pour sortir d’un programme.



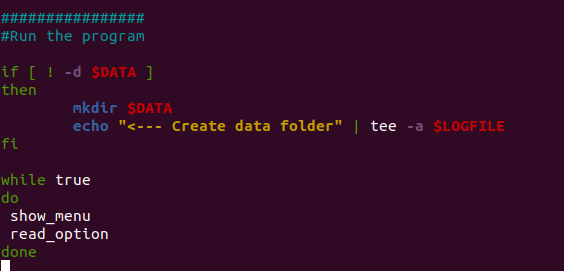
Read\_option() :

Cette commande permet de lire ce que l’utilisateur a écrit. La commande CASE permettra d’aiguiller les options vers les commandes appropriées.

Exemple : Écrire le nombre 6 permettrait d’appeler la fonction viewLog.

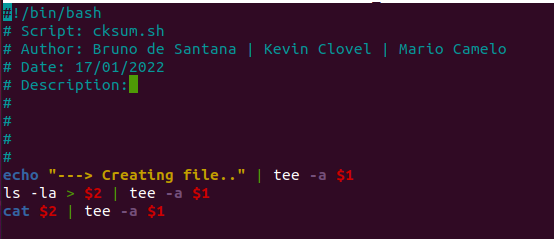


Ici nous avons un test qui valide si le répertoire existe, si ce n’est pas la cas le répertoire sera créé.

Et pour finir, la boucle WHILE où l’on exécute la fonction show\_menu et read\_option. C’est une boucle infini qui se terminera lorsque l’utilisateur choisira l’option de sortie. 

### CREATE\_FILE.SH

Un script de création de fichier.



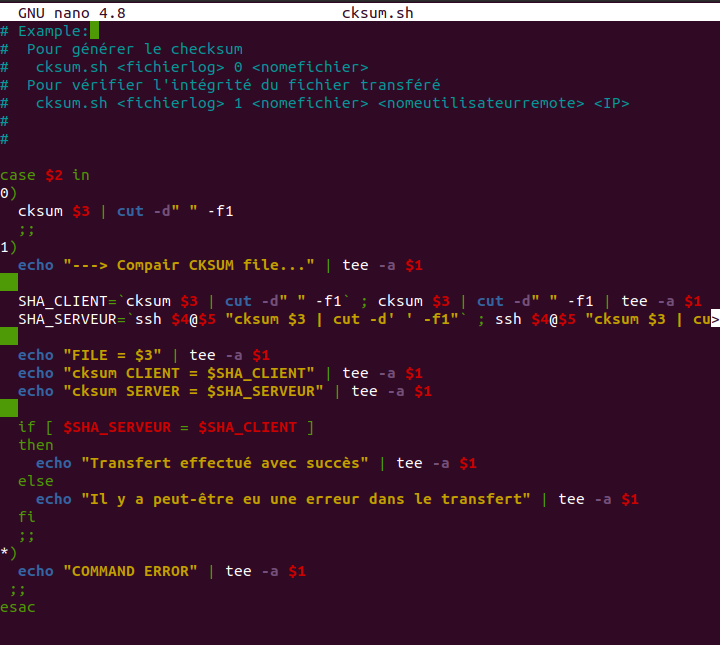
### VALIDE.SH

Script avec une boucle IF afin de valider l’existence du fichier afin de savoir si c’est un fichier régulier ou un répertoire.



### CKSUM.SH

Ce script possède 2 fonctions, la première permettra de générer un cksum, et l’autre d’aller sur le SERVEUR avec la commande SSH, et de prendre le nombre que le cksum aura attribuer au fichier du serveur. Après validation, le serveur et le client possèderont le même nombre crée par le cksum.



### SEND.SH

Le script send.sh utilise la command SCP pour envoyer le fichier du client vers le serveur.

