

BUT Informatique

# SAE1.03

## Installation d'un poste pour le développement

Travaux dirigés

MEZGHICHE Ahmed, OURRAD Sami, JOMIE Antonin  
27/11/2023

## Sommaire

### Table des matières

<b>#1 Environnement pour le dev .....</b>	<b>2</b>
<b>#2 Infrastructure de test .....</b>	<b>6</b>
<b>#3 Infra ‘sans les mains’ .....</b>	<b>9</b>

## SAE1.03 – Installation d'un poste pour le développement / Compte rendu

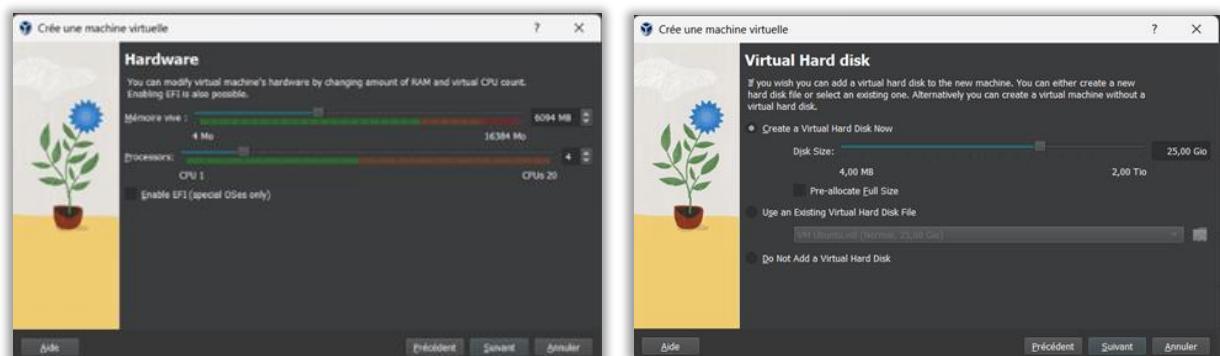
### • Installation d'un OS Linux

Pour installer une machine virtuelle sous le système d'exploitation Ubuntu, nous avons dû configurer certains paramètres lors de sa création sur le logiciel VirtualBox qui est un logiciel libre de virtualisation :

Nous avons débuté par insérer le CD virtuel Ubuntu LTS grâce au fichier .ISO, téléchargé sur le site officiel de ce dernier et ensuite nommé le nom de la machine virtuelle pour enfin configurer son mot de passe.



Ensuite nous lui avons alloué l'occupation de la RAM (6 Go), CPU (4 cœurs), et du disque dur virtuel souhaité (25 Go) (selon la configuration du PC physique).

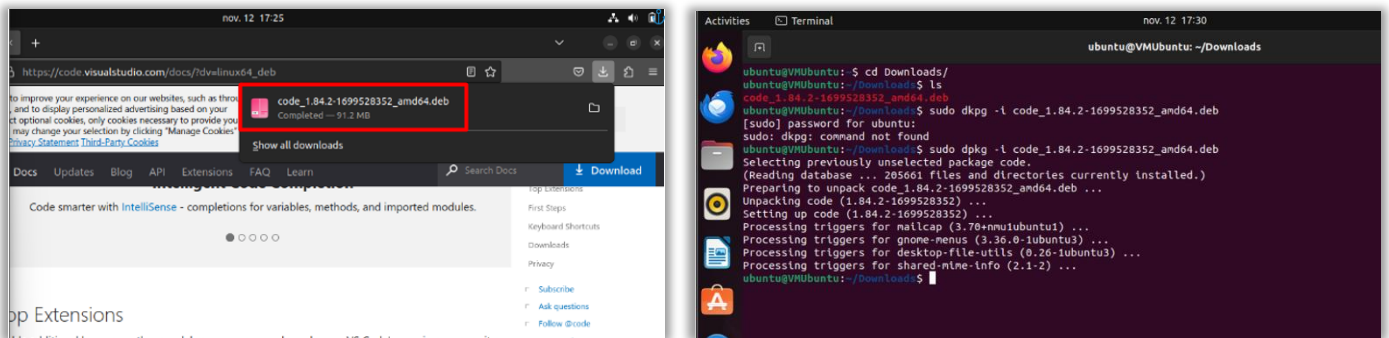


## • Outils de développement

### 1. Installation de Visual Studio Code/Google Chrome

---

Pour pouvoir installer Visual Studio Code et Google Chrome, nous avons téléchargé leur fichier .deb à travers le navigateur FireFox. Ensuite nous avons procédé à l'installation à partir du terminal de la machine virtuelle.



### 2. Installation de Python3, Git

---

En suivant le même procédé sur l'invite de commande, nous avons installé Git et Python3 en rentrant une ligne de code spécifique :

- `sudo apt update` (met à jour la base de donnée locale de paquets disponible)
- `sudo apt install python3 git` (installe l'interpréteur Python/ et également Git)

## • Environnement de Travail

### 1. Accès au Réseau Physique Depuis l'hôte

---

Pour pouvoir avoir accès au réseau physique depuis l'hôte nous avons eu à modifier une configuration de la VM dans l'interface VirtualBox, dans la section Réseau, nous avons modifié le mode d'Accès Réseau (qui était au départ en NAT) en mode Accès Par Pont. Ainsi nous permettons à la machine d'avoir une adresse IP directement sur le même réseau que l'hôte)

### 2. Contrôle depuis l'hôte (MobaXterm)

---

Pour avoir un contrôle de la machine Ubuntu depuis l'hôte, nous avons à tout d'abord installer le Server SSH via son terminal à partir de cette commande :

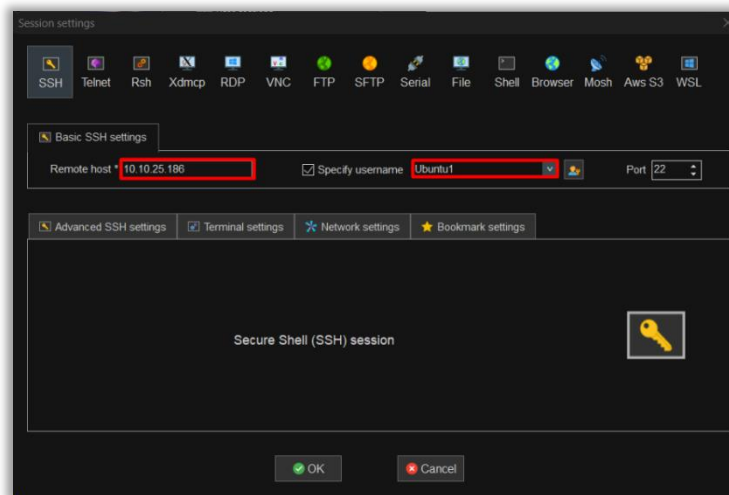
- `sudo apt install openssh-server` (Installe le serveur OpenSSH sur le système)

Nous devons également récupérer l'adresse IP de la machine virtuelle par cette commande :

- `ip a`

En l'occurrence, l'IP de la machine est 10.10.25.186


Ensuite, lorsque nous nous rendons sur le logiciel MobaXTerm, nous lançons une nouvelle session SSH en rentrant ces données spécifiées :

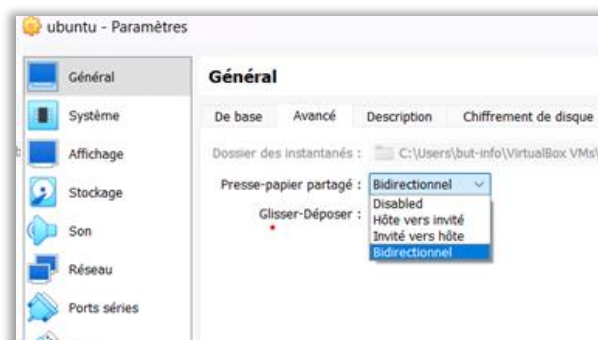


Lorsque nous lançons la session SSH, nous rentrons le mot de passe de la machine Ubuntu, et nous pouvons enfin contrôler la machine depuis l'hôte et accéder à ses fichiers.



- **Transfert de fichiers avec l'hôte**


### 3. Drag & Drop

Pour autoriser le Drag & Drop, c'est plutôt simple, il suffit juste d'aller dans les paramètres généraux de la machine virtuelle, aller dans  Général ensuite dans avancé et choisir « Bidirectionnel ». Ce qui va permettre de faire des copier-coller entre l'hôte et la machine virtuel



## 4. Dossier Partagé

Dans la fenêtre de configuration de la machine virtuelle, nous avons eu à accéder dans la rubrique des dossiers partagé, puis cliquer sur ce logo  afin d'ajouter le chemin d'accès du dossier commun entre la machine virtuelle et l'hôte, en l'occurrence : (« C:\Users\but-info\Documents\SAE1.03\Dossier Partagé »). Cependant, ce n'est pas uniquement la manipulation à réaliser. Une fois que nous avons attaché le chemin d'accès à la machine, nous la lançons et nous lui insérons une image CD par cette fonctionnalité :  Insérer l'image CD des Additions invité...

Ensuite nous ouvrons le disque optique  **VBox\_GAs\_6.1.36** Et nous exécutons le fichier Autorun.sh en tant que programme. Enfin, nous rentrons cette succession de commande pour finaliser le dossier partagé :

- `sudo apt update`
- `sudo apt-get install -y build-essential linux-headers-$(uname -r)`
- `sudo apt-get install virtualbox-guest-utils`
- `sudo adduser $(whoami) vboxsf` (donne à l'utilisateur l'accès aux partages de fichiers VirtualBox sur le système)

## 5. Services réseau (FTP, SFTP)

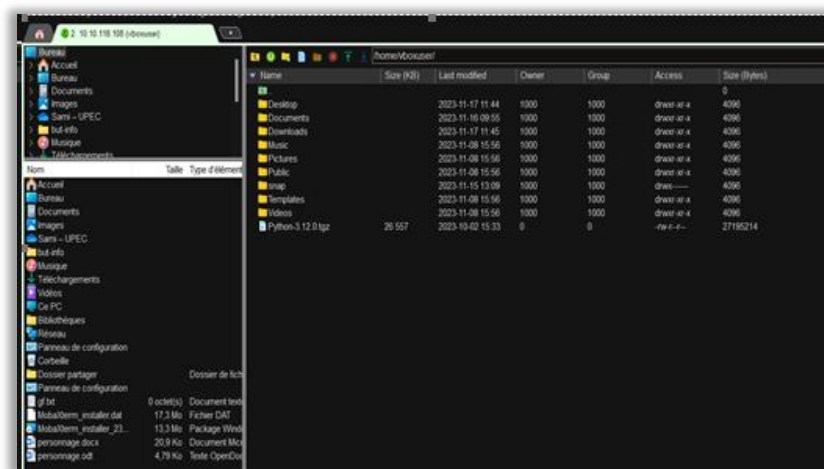
### • Pour le FTP

Dans le terminal de la machine virtuelle, nous débutons par rentrer ces commandes :

- `sudo apt install vsftpd` (installe le serveur FTP sur Ubuntu)

Ensuite dans MobaXterm ou un équivalent (Filezilla...), nous lançons une session FTP :

Nous insérons l'adresse IP (elle se récupère avec la commande `ip a`) de la machine Ubuntu avec le port qui convient (pour un serveur ftp on prend port 21) :

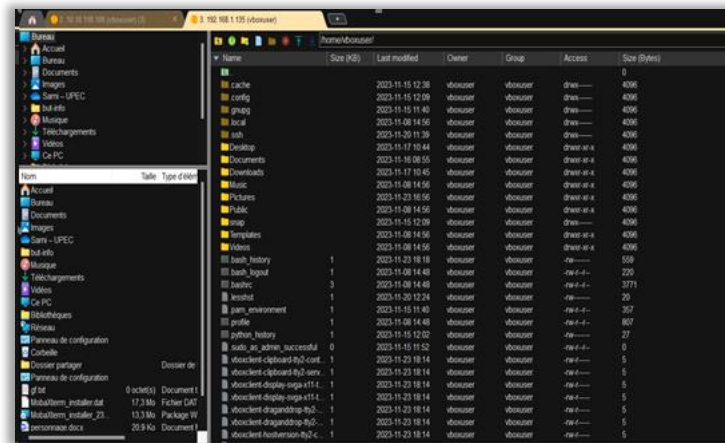


- **Pour SFTP :**

Nous entamons la même procédure avec de différentes commandes :

- `sudo apt install openssh-client` (télécharge et installe le client OpenSSH)
- `sudo systemctl status ssh` (vérifie l'état actuel du service SSH)

On se rend ensuite sur MobaXterm ou un autre dérivé et nous nous connectons avec l'adresse IP de la machine en lançant une session SFTP :



## Fin de la 1<sup>ère</sup> Partie .

# #2 Infrastructure de test

## 1. Installation d'un OS Linux

---

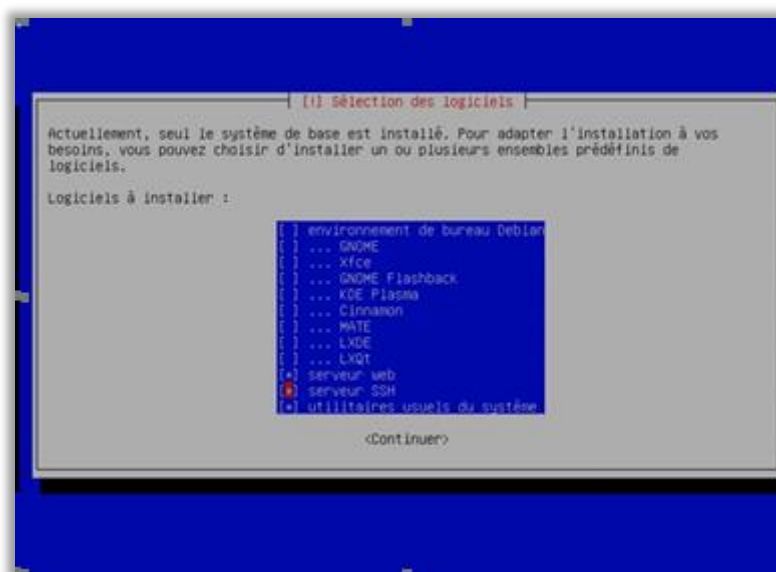
- **Debian**

Nous installons une VM de façon normale mais on utilise comme ISO « Debian » téléchargeable ici : <https://www.debian.org/index.fr.html>



Arrivé dans la page de Debian nous cliquons sur **Install** :






Nous installons normalement Debian jusqu'à arriver sur cette fenêtre :





On décoche  environnement de bureau Debian (qui est l'interface graphique) et  ... GNOME

 serveur web  
 serveur SSH  
 utilitaires usuels du système

Et on coche les 3 ligne suivante :

**Serveur web** : Permet d'allouer un serveur à la machine

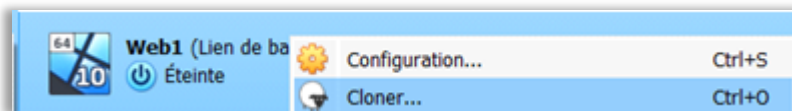
**Serveur SSH** : Permet de mettre un serveur que l'on peut contrôler à distance et de ne pas l'installer dans le terminal (CMD)

**Utilitaires usuels du système** : Va permettre d'être sur le cmd dès le début sans interface graphique

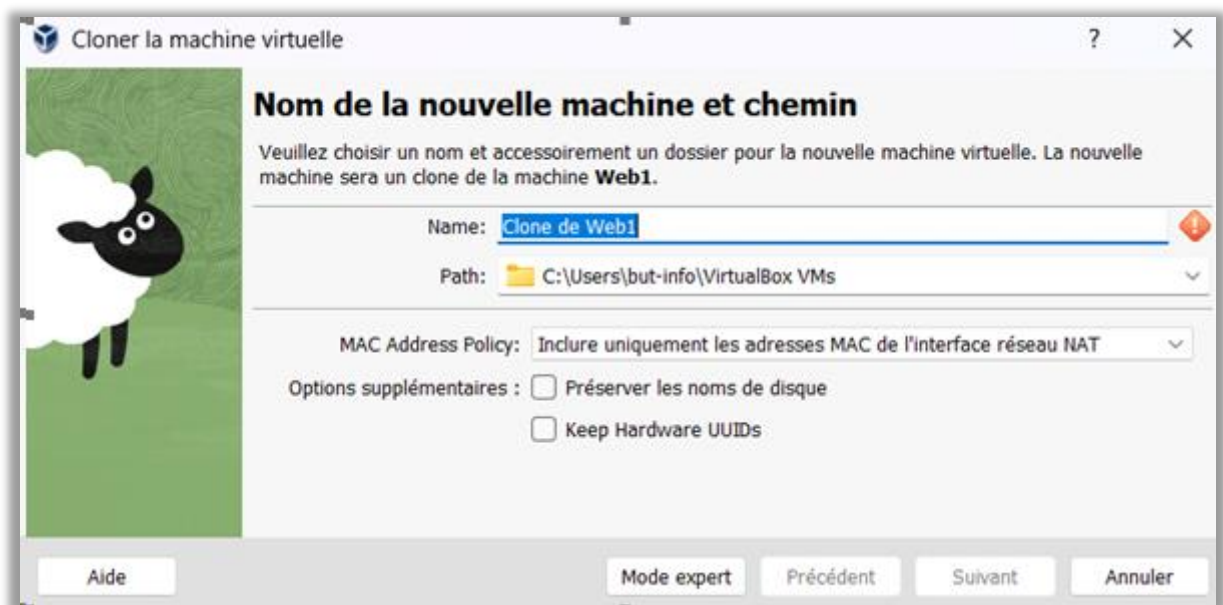
- **Création d'un 'pool' de serveurs**

## 1. Cloner

Il est assez simple de cloner une machine , il suffit juste de faire un clic droit sur la machine que nous voulons cloner et on clique sur « Cloner... »



Nous lui donnons ensuite le nom souhaité nous le clonons intégralement à la VM de base :

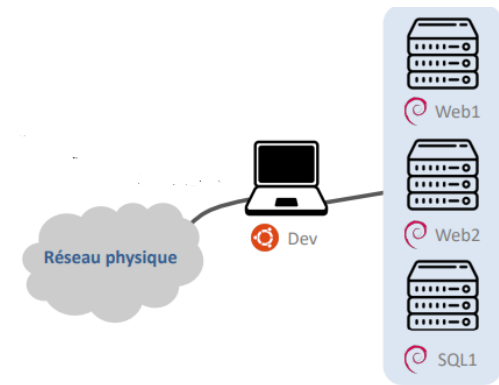


Ainsi, le clonage est réalisé.

### • Accès réseau uniquement depuis le poste dev

Pour avoir un accès réseau uniquement depuis le poste dev, nous avons tout d'abord simulé un pool de serveur auparavant avec 1 poste dev (Ubuntu) et 3 « sous-machines » (Debian).

**Schématisation du pool de serveur :**



- Pour lier le réseau physique et le poste dev nous avons inséré une carte réseau en mode « Accès par pont »
- Pour lier le poste dev au 3 machines virtuelle nous avons inséré une deuxième carte réseau en mode Réseau Privé hôte dans cette dernière, tandis que pour les 3 « sous-machines » nous leur avons alloué une seule carte réseau en mode « Réseau Privé Hôte »
- Ainsi le pool de serveur est schématisé

### Test de la communication :

On peut s'assurer que la communication est bien effectuée avec un : ping « l'adresse IP de la VM en Réseau privé hôte »

```
vboxuser@ubuntu:~$ ping 192.168.56.105
PING 192.168.56.105 (192.168.56.105) 56(84) bytes of data:
 64 bytes from 192.168.56.105: icmp_seq=1 ttl=64 time=1.03 ms
 64 bytes from 192.168.56.105: icmp_seq=2 ttl=64 time=1.27 ms
 64 bytes from 192.168.56.105: icmp_seq=3 ttl=64 time=0.964 ms
 64 bytes from 192.168.56.105: icmp_seq=4 ttl=64 time=1.94 ms
```

S'il y a bien des échanges alors c'est bon, les VM sont connectées entre elles

Si cela ne marche pas, alors il y aura un message d'erreur :

```
ubuntutest@ubuntutest-VirtualBox:~$ ping 192.485.15.2
ping: 192.485.15.2: Échec temporaire dans la résolution du nom
ubuntutest@ubuntutest-VirtualBox:~$
```

Fin de la 2<sup>ème</sup> Partie

# #3 Infra 'sans les mains'

## 1. Création d'un Run

---

Nous commençons par créer un préfixe, ici, nous utiliserons le mot 'valeur' qui va s'incrémenter de 1 à chaque fois que nous lancerons l'exécutable:

```
@echo off

REM Incrémentation de la run
setlocal enabledelayedexpansion

set "fichierValeur=valeur.txt"

rem Lire la valeur actuelle depuis le fichier
set /p valeur=<%fichierValeur%

rem Incrémenter la valeur
set /a valeur+=1

rem Enregistrer la nouvelle valeur dans le fichier
echo %valeur% > %fichierValeur%

echo Lancement de la Run (Nombre de Pool crée : %valeur%)
```

Pour cette création de Run on a utiliser le codage en .batch (qui est beaucoup plus simple du fait que cela soit juste les commandes équivalentes du Terminal)

Voici le contenu du code au lancement de l'exécutable:

```
echo Lancement de la Run (Nombre de Run crée : %valeur%)

VBoxManage clonevm "Web" --name="Run%valeur%-Web1" --register --mode=all --options=keepallmacs --options=keepdisknames --options=keepuuids
VBoxManage clonevm "Web2" --name="Run%valeur%-Web2" --register --mode=all --options=keepallmacs --options=keepdisknames --options=keepuuids
VBoxManage clonevm "SQL1" --name="Run%valeur%-SQL1" --register --mode=all --options=keepallmacs --options=keepdisknames --options=keepuuids
```

Figure 1: Le programme clone respectivement le pool de serveur original, créé manuellement, pour chaque clonage nous lui attribuons le numéro de la run avec la variable %valeur%

Ensuite, nous demandons à l'utilisateur combien veut-il de ram et de CPU de manière respective pour chaque machine clonés. Voici un exemple du code pour la première machine

```

REM Première Machine

echo (1ère Machine) Saisissez la ram (entre 4Mo et 6000Mo):
set /p Ram1=

echo (1ère Machine) Saisissez le nombre de CPU à allouer (entre 1 et 4) :
set /p CPU1=

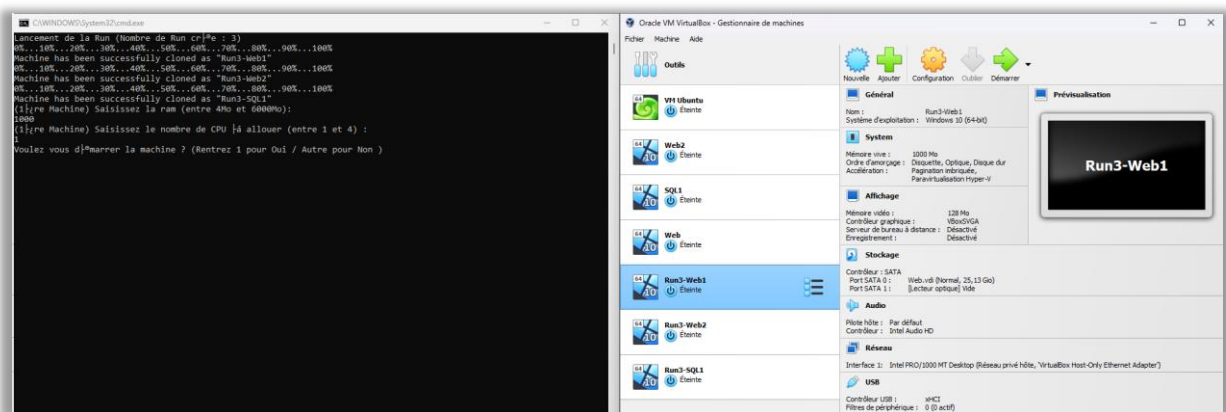
VBoxManage modifyvm "Run%valeur%-Web1" --memory %Ram1%
VBoxManage modifyvm "Run%valeur%-Web1" --cpus %CPU1%

echo Voulez vous démarrer la machine ? (Rentrez 1 pour Oui / Autre pour Non )
set /p Launch1=

if %Launch1% equ 1 (
    echo Lancement en cours de la machine
    VBoxManage startvm "Run%valeur%-Web1"
) else (
    echo Lancement de la machine annulé
)

```

Figure 2 : Nous initialisons les variable Ram1 et CPU1 dont leur valeur sera inséré par l'utilisateur, pour ensuite, allouer ces valeurs à la machine cloné à l'aide de la commande vboxmanage. Et enfin nous demandons à l'utilisateur d'entrer 1 si il souhaite lancer la machine après son clonage, et d'entrer une autre valeur si il ne le souhaite pas



Ainsi, nous pouvons bien voir par exemple que le programme fut bien exécuté, car nous pouvons apercevoir que la Run numéro 3 (étant donnée que l'utilisateur a exécuté le programme 3 fois, et a supprimé les 2 premières à l'aide d'un autre programme que nous verrons à la fin) vient d'être créée, et nous pouvons notamment voir que la ram entrée par l'utilisateur pour la première machine a bien été allouée.

## 2. Screenshot Régulier/ Sauvegarde Régulière

```
@echo off
setlocal enabledelayedexpansion

echo Entrez le numéro de la run à laquelle vous voulez screenshot régulièrement :
set /p run=

echo Combien de fois voulez vous screenshot?
set /p nombre=

VboxManage startvm "Run%run%-Web1"
VboxManage startvm "Run%run%-Web2"
VboxManage startvm "Run%run%-SQL1"

for /L %%i in (1,1,%nombre%) do (
    vboxmanage controlvm "Run%run%-Web1" screenshotpng "C:\Users\but-info\Documents\SAE1.03\Screenshot\capture00%%i.png"
    vboxmanage controlvm "Run%run%-Web2" screenshotpng "C:\Users\but-info\Documents\SAE1.03\Screenshot\capture00%%i.png"
    vboxmanage controlvm "Run%run%-SQL1" screenshotpng "C:\Users\but-info\Documents\SAE1.03\Screenshot\capture000%%i.png"
    TIMEOUT /T 300
)
```

Pour pouvoir réaliser un screenshot régulier, nous demandons à l'utilisateur quel est la run qu'il souhaite capturer régulièrement et combien de capture veut-il.

La structure de contrôle «for » est en réalité une boucle qui s'incrémente de 1 jusqu'au nombre souhaité, et va permettre à l'utilisateur de screenshot automatiquement ses machines.

La commande « TIMEOUT /T 300 » sert ainsi à respecter le délai de 5 minute demandé.

```
@echo off
setlocal enabledelayedexpansion

echo Entrez le numéro de la run à laquelle vous voulez sauvegarder régulièrement :
set /p run=

echo Combien de fois voulez vous sauvegarder?
set /p nombre=

VboxManage startvm "Run%run%-Web1"
VboxManage startvm "Run%run%-Web2"
VboxManage startvm "Run%run%-SQL1"

for /L %%i in (1,1,%nombre%) do (
    vboxmanage snapshot "Run%run%-Web1" take Sauvegarde00%%i
    vboxmanage snapshot "Run%run%-Web2" take Sauvegarde00%%i
    vboxmanage snapshot "Run%run%-SQL1" take Sauvegarde000%%i
    TIMEOUT /T 300
)
```

Nous suivons ainsi le même procédé pour la sauvegarde régulière dans un autre, sauf que dans la boucle « for », la commande permettant de screenshot est ainsi remplacé par celle permettant de sauvegarder les machines

### 3. Destruction d'un Run

---

```
@echo off
setlocal enabledelayedexpansion

echo Entrez le numéro de la run que vous voulez détruire :
set /p run=

vboxmanage unregistervm "Run%run%-Web1" --delete
vboxmanage unregistervm "Run%run%-Web2" --delete
vboxmanage unregistervm "Run%run%-SQL1" --delete

echo Run détruite avec succès|
```

Pour détruire la Run souhaité, nous demandons à l'utilisateur d'insérer le numéro de la run, et ensuite à l'aide de la commande vboxmanage, nous supprimons les 3 clones créé selon le numéro de la run.

Un message s'affichera ensuite pour annoncer que la run a bien été détruite.

<b>Fin de la 3<sup>ème</sup> Partie</b>
---