

Contents

1	Installation de la VM	2
1.1	Questions OS :	2
1.2	Préparation du Système	3
1.2.1	Installation des suppléments invités:	4
2	À propos de la distribution Debian	5
2.1	Documentations	5
2.2	Quelques Questions	5
3	Installations paquets et configuration	7
3.1	Qu'est-ce que gitk ?	8
3.1.1	Qu'est-ce que git-gui ?	10
4	Gitea	11
4.1	QUESTIONS	11
4.2	Installation gitea:	11
4.2.1	Port-forwarding sur VM:	11
4.2.2	Pré-requis	11
4.2.3	Téléchargements	11
4.2.4	Vérifications des fichiers téléchargés avec GPG	11
4.3	Préparer l'environnement Gitea	12
4.3.1	Version Git	12
4.3.2	Utilisateur Git	12
4.4	Lancer Gitea	12
4.4.1	Configurer Gitea	13
4.4.2	Protéger les fichiers Gitea	13
4.5	Lancer Gitea après installation	13
4.6	Version Gitea:	14
4.7	Installer une version spécifique de Gitea:	14
4.8	Utilisation de Gitea:	14

1 Installation de la VM

“64-bit” dans “**Debian 64-bit**” fait référence à l’architecture du processeur et du système d’exploitation. Plus précisément, ça signifie que la version de Debian est conçue pour fonctionner sur un processeur **64 bits**, qui peut traiter des instructions et gérer la mémoire de manière plus efficace qu’un processeur **32 bits**.

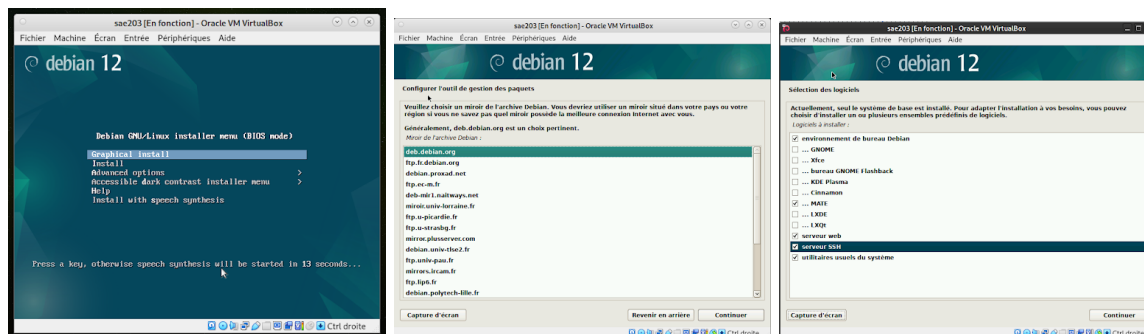
Un OS avec une architecture 32 bits ne pourra pas traiter autant de données qu’un OS fonctionnant sur du 64 bits. Les bits font références à un type d’architecture informatique dans lequel le processeur et le système d’exploitation peuvent traiter des données par **morceaux de 3/64 bits à la fois**. Cela signifie que l’ordinateur peut traiter des données et effectuer des calculs sur des chiffres de **32/64 bits de long**.

A - Configuration réseau utilisée par défaut : Intel Pro/1000 MT Desktop (NAT)

B - Nom du fichier XML contenant la configuration de la VM : sae203.vbox

C - On peut changer le nombre de processeurs dans le fichier sae203.vbox en l’ouvrant avec nano dans un terminal.

Installation de l’OS:



1.1 Questions OS :

Un fichier ISO amorçable est conçu pour **s’exécuter lorsque vous démarrez votre PC**. Les exemples sont nombreux, mais l’un d’entre eux concerne l’**utilisation d’un fichier ISO** pour installer un système d’exploitation.

MATE est un fork de **GNOME 2**. Il fournit un environnement de bureau attractif et intuitif en se basant sur les métaphores traditionnelles pour GNU/Linux et d’autres systèmes d’exploitation similaires à Unix.

Chaque fois que vous visitez un site internet, vous vous adressez à un **serveur web** pour récupérer la page que vous souhaitez consulter. Le serveur répond à votre requête en envoyant la page à votre ordinateur ou appareil avant que vous ne puissiez la consulter dans votre navigateur web (aussi appelé client web).

Secure Shell (SSH) est un protocole de communication sécurisé. Le protocole de connexion impose un échange de clés de chiffrement en début de connexion. Par la suite, tous les segments TCP sont authentifiés et chiffrés. Il devient donc impossible d’utiliser un analyseur de paquets (sniffer) pour voir ce que fait l’utilisateur. Le protocole SSH a été conçu avec l’objectif de remplacer les différents protocoles non chiffrés comme rlogin, telnet, rcp et rsh.

Un **serveur mandataire** ou **proxy** (de l’anglais) est un serveur informatique qui a pour fonction de relayer des requêtes entre un poste client et un serveur. Les serveurs mandataires sont notamment utilisés pour assurer les fonctions suivantes :

- la mémoire cache
- la journalisation des requêtes (“logging”)
- la sécurité du réseau local
- le filtrage et l’anonymat

L’utilité des serveurs mandataires est importante, notamment dans le cadre de la sécurisation des systèmes d’information.

1.2 Préparation du Système

1) Mettre les droits Sudo à user :

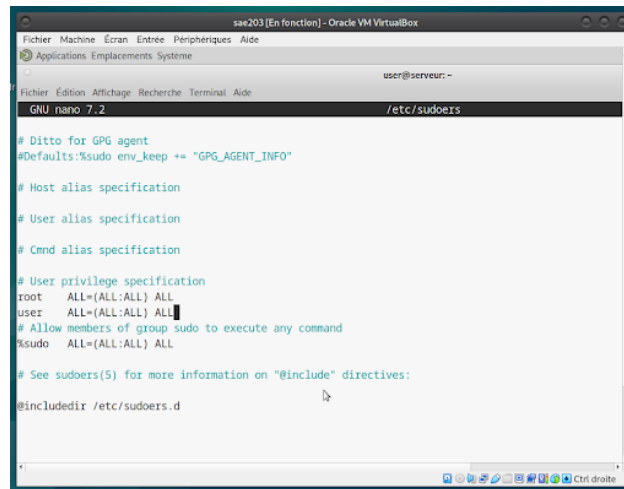


Figure 1: Commandes Sudo

2) Installation des additions guest :

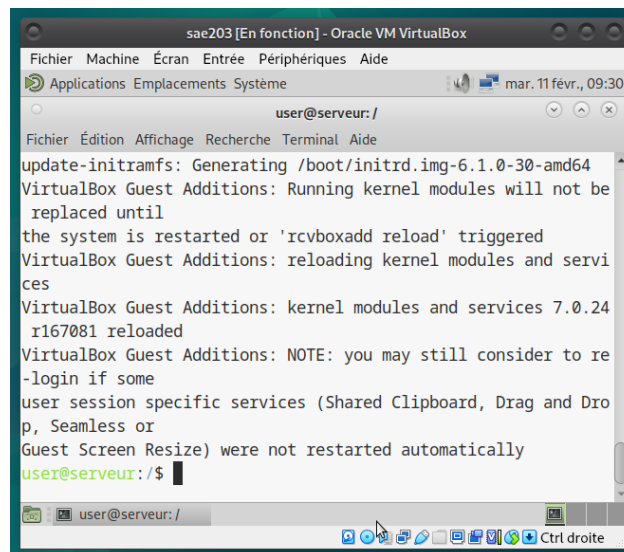


Figure 2: additionsGuest

La commande pour savoir à quel groupe est assigné l'utilisateur est : `groups nom_utilisateur`

La commande `uname -r` permet de regarder la version du noyau linux. (Celle de la VM est 6.1.0-31-amd64.)

Les **suppléments invités** (Guest Additions) sont un ensemble de pilotes et d'outils fournis par VirtualBox pour améliorer l'intégration entre l'hôte et la machine virtuelle (VM).

Elles permettent :

- Une amélioration des performances et de l'affichage avec une meilleure résolution dynamique.
- Une meilleure interaction entre l'hôte et la VM avec entre autre le partage du presse-papiers et la permission de glisser-déposer de fichiers.

1.2.1 Installation des suppléments invités:

Pour installer les suppléments invités il faut insérer le disque des dépendances dans le menu périphérique puis le monter avec la commande **mount**.

La commande mount: Elle sert à monter un **système de fichiers** (partition, clé USB, disque dur, ISO...) afin qu'il soit accessible dans l'arborescence du système.

Ici, la commande **Pour Installer ces outils:**

- **sudo mount /dev/cdrom /mnt** *Pour monter les fichiers sur votre appareil.* - **sudo /mnt/VBoxLinuxAdditions.run**
Pour exécuter le script d'installation des outils. - **sudo reboot** *Pour redémarrer le système.*

2 À propos de la distribution Debian

2.1 Documentations

- <https://www.debian.org/doc/>
- <https://debian-fr.org>
- <https://forum-debian.fr>
- <https://debian-facile.org>

2.2 Quelques Questions

Le nom [Debian](#) vient de Debian = Debra + Ian Lan est le créateur de Debian distribution. Debra est la copine de Lan. [Qu'est ce que Debian ?](#)

Le Projet Debian est une association de personnes qui ont fait cause commune afin de créer un système d'exploitation libre.

Il s'efforce de fournir à tous ses utilisateurs une documentation facilement accessible et qui est un système d'exploitation libre de haute qualité englobe les manuels techniques qui décrivent le fonctionnement et l'utilisation des programmes.

Il existe 3 durées de prise en charge (support) de ces versions, dont :

- La durée minimale ; Une durée de vie totale d'environ 3 ans (entre sa publication et la fin des mises à jour de sécurité).
- La durée en support long terme ([LTS](#)) : Debian Long Term Support, pour prise en charge à long terme, est un projet pour étendre la durée de vie de toutes les versions stables de Debian à (au moins) 5 ans.
- La durée en support long terme étendue ([ELTS](#)) : La prise en charge à long terme étendue ([ELTS](#)) est une offre commerciale offrant un prolongement de la durée de vie de 10 ans des versions de Debian. Elle a une structure de 32 bits et ne pourra pas traiter autant de données qu'un OS fonctionnant sur du 64 bits. Les bits font références à un type d'architecture informatique dans lequel le processeur et le système d'exploitation peuvent traiter des données par morceaux.

Les versions successives de la distribution Debian portent à la fois un numéro de version traditionnel et des noms de code tirés des personnages du film Toy Story de Pixar/Disney (1995).

Dès que de nouveaux bogues de sécurité sont décelés dans les paquets, les responsables Debian et les auteurs amont les corrigent généralement dans les journées ou les heures suivantes.

Une fois le bogue résolu, un nouveau paquet est fourni sur [Debian Security](#).

[Faire une mise à jour de sécurité Debian](#)

Versions courantes activement maintenues par debian :

- oldstable - La version précédant la version stable (Bullseye).
- stable - La version stable actuelle (Bookworm).
- testing - La prochaine version stable (Trixie).
- unstable - La version de développement instable (Sid), c'est là que sont ajoutés les nouveaux paquets ou les nouvelles mises à jour.

À tout moment, il y a une version stable de Debian qui est maintenue par l'équipe de sécurité Debian. Quand une nouvelle version stable sort, l'équipe de sécurité maintient la version précédente pendant un an tandis qu'elle maintient aussi la nouvelle version courante. Seule la version stable est recommandée pour un environnement de production.

Et donc Debian maintient trois versions activement : la version stable stable, la version unstable et la version testing.

[Les versions de Debian](#)

Chaque distribution majeur possède un **nom de code** différent.

Les noms de code des versions de Debian sont inspirés des personnages du film d'animation "Toy Story". Cette tradition a été instaurée par Bruce Perens, qui était responsable du projet Debian à l'époque et travaillait

chez Pixar, le studio derrière “Toy Story”. Par exemple la branche “unstable” est nommée “Sid” , la branche Old Stable à le nom de code “Bullseye” (auparavant “woody” le personnage principal du film) et la Branche testing s’appelle “Trixie”.

- [Distributions Debian](#)
- [FAQ Debian](#)

L’un des atouts de Debian fut le nombre d’architecture officiellement prises en charge.

Par exemple, la version Bullseye prend en charge neuf architectures au total, dont les suivantes :

- **amd64** pour AMD64 PC 64 bits / Intel EM64T / x86-64
- **i386** pour i386 PC 32 bits / Intel IA-32
- **ppc64el** pour PowerPC 64 bits little-endian Motorola/IBM PowerPC
- **s390x** pour S390 64 bits IBM S/390
- **armel** pour ARM
- **armhf** pour les anciens matériels ARM et de plus récentes architectures 32 bits
- **arm64** pour les architectures ARM 64 bits Arch64
- **mipsel** pour MIPS 32 bits little-endian
- **mips64el** pour MIPS 64 bits little-endian

(Source : <https://wiki.debian.org/fr/DebianBullseye#Architectures>)

Le premier nom de code utilisé fut buzz (Debian 1.1), c’est le cosmonaute Buzz Lightyear.

Cette version a été publiée le 17 juin 1996 sous le nom de Debian GNU/Linux 1.1.

Cette version était totalement en ELF (Executable and Linkable Format), utilisant le noyau Linux 2.0.

Le derrière nom de code annoncé à ce jour est sid, le garçon des voisins qui casse tous les jouets dans Toy Story.

Sid est en phase de développement, il n’y a donc pas de date de sortie annoncée.

La version précédente, le bookworm est sortie le 10 juin 2023.

A savoir que Debian Sid n’a pas de numéro de version, car c’est une branche en perpétuel développement.

Contrairement aux versions stables de Debian (comme [Debian 12 “bookworm”](#)).

3 Installations paquets et configuration

Pour remplacer la chaîne @@UUID@@ par un identifiant unique universel.

```
sed -i -E "s/(--iprt-iso-maker-file-marker-bourne-sh).*$/\1=$(cat /proc/sys/kernel/random/uuid)/" S203-Debian12.viso
```

Modifier votre configuration (et recommencez l'installation) afin de :

- Ajouter le droit d'utiliser sudo à l'utilisateur standard
- Installer l'environnement MATE.
- Ajouter les paquets suivants :
- sudo : sinon la gestion sudo est inutile
- git, sqlite3, curl : pour préparer l'installation de la semaine prochaine
- bash-completion : va vous simplifier grandement l'écriture des lignes de commande
- neofetch : pas très utile mais c'est un classique dans son genre (essayez-le)

META PAQUETAGES :

Pour installer ces packets automatiquement, on ajoute ceci : "preseed-fr.cfg" ce qui nous donne : `d-i pkgssel/include string sudo git sqlite3 curl bash-completion neofetch`

Pour installer l'environnement, on ajoute également :

- Pour l'environnement Debian et MATE: `d-i taskssel/first multiselect standard, mate-desktop`
- Pour le serveur WEB/SSH `d-i pkgssel/include string apache2 openssh-server`
- Pour mettre les droits sudo : `d-i passwd/user-default-groups string sudo`

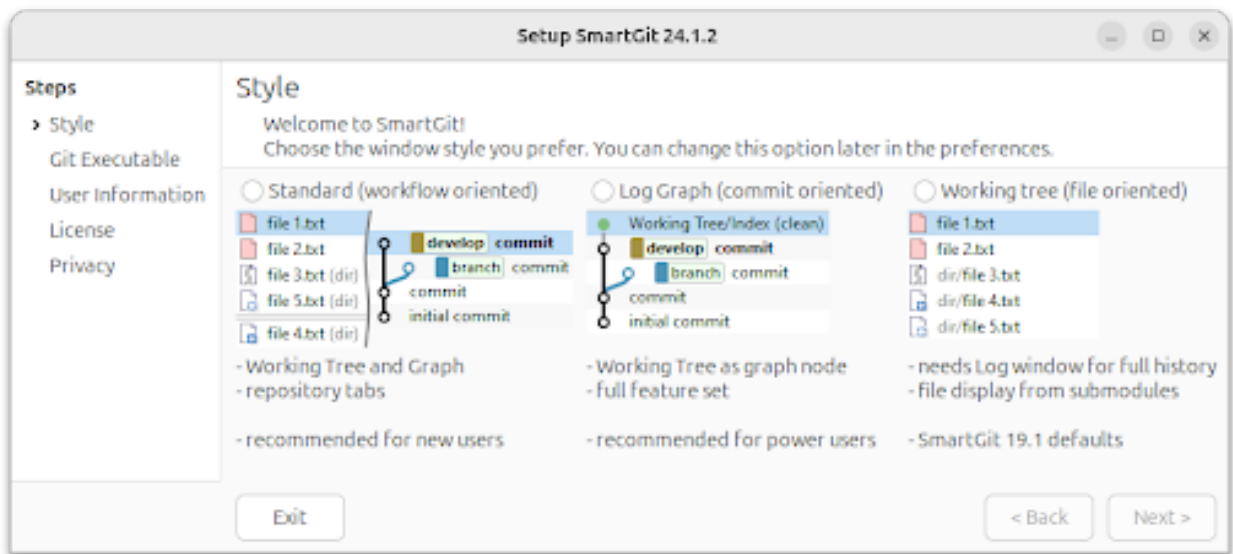
Installation d'un GUI git (Graphical User Interface) Smart Git

Installer les fichiers d'installation depuis le site [syntevo](https://syntevo.com/projects/smartgit/).

Extraire le fichier avec la commande : `tar xzf smartgit-linux-24_1_2.tar.gz`

Puis lancer le fichier d'installation avec : `./bin/smartgit.sh`

Vous devriez arriver sur cette fenêtre :



Suivez l'installation du programme jusqu'à arriver sur la dernière page et cliquez sur **Finish**

Le programme se lancera :

Les trois boutons en haut permettent de créer, cloner et rechercher un répertoire Git.

Vous verrez aussi vos repository récents.

Voici quelques captures d'écran en utilisant quelques fonctionnalités :

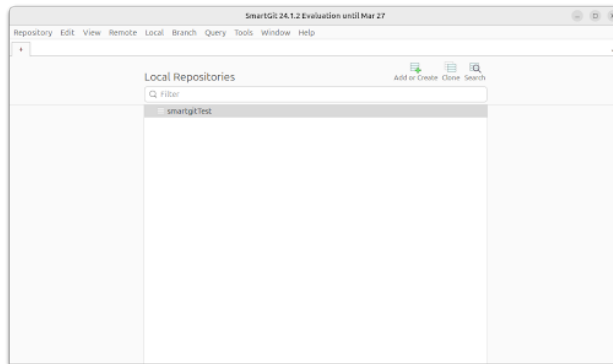


Figure 3: SmartGit2

A. Ajout d'un fichier et inclure la modification dans un commit :

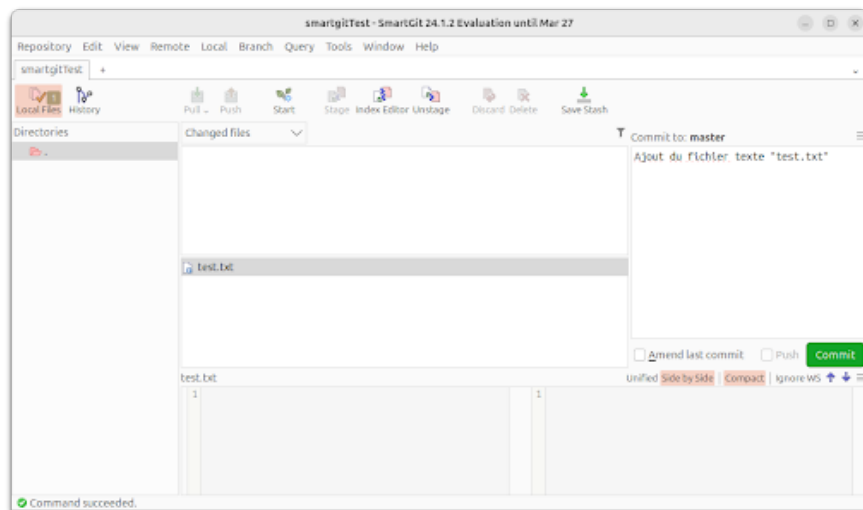


Figure 4: AjoutFichier

B. De même avec du contenu supplémentaire dans le fichier, on peut voir les changements :

C. Voir l'historique de tous les changements :

D. Pour push il faut d'abord ajouter le serveur où vos fichiers vont être envoyé :

E. Connectez-vous, identifiez vous avec votre compte et *push* les *commit*, voici le résultat :

Pour lancer l'application depuis le terminal faites un lien symbolique avec le smartgit.sh dans le **bin** à l'aide de `ln -s`.

Placez le dans `/usr/local/bin`. Vous pouvez aussi l'ajouter au menu grâce au script nommé `add-menuitem.sh`.

3.1 Qu'est-ce que gitk ?

gitk est une interface graphique légère pour visualiser l'historique d'un dépôt Git. C'est l'un des outils officiels fournis avec Git.

Il permet de voir :

- Les commits sous forme de graphe
- Les branches et fusions
- Les différences entre versions
- Les auteurs et messages de commit

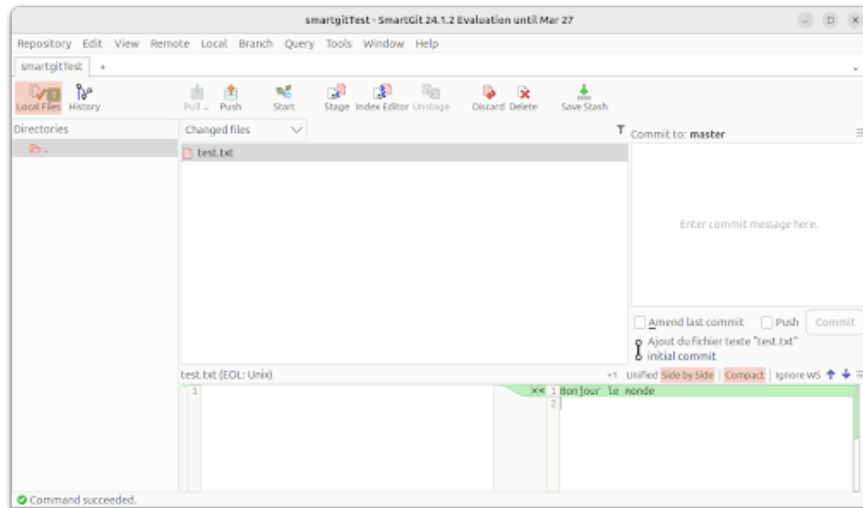


Figure 5: AjoutFichier2

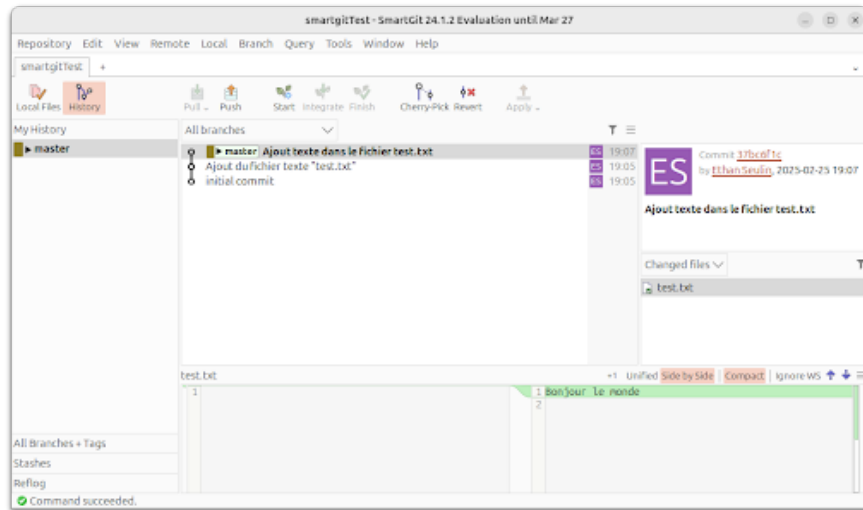


Figure 6: VoirHistorique

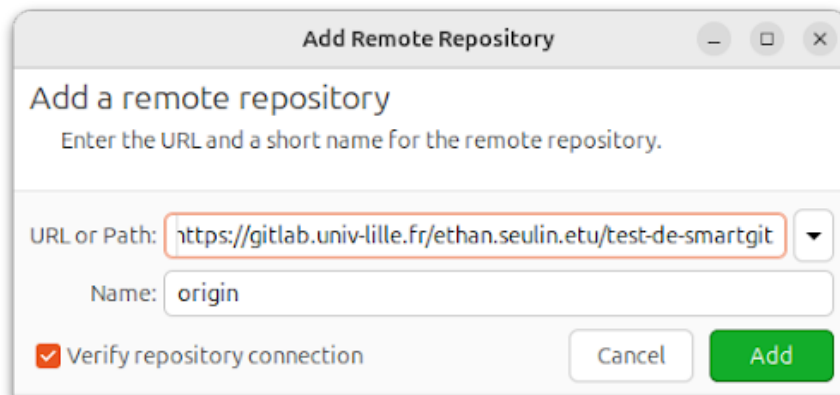


Figure 7: AjoutServeur

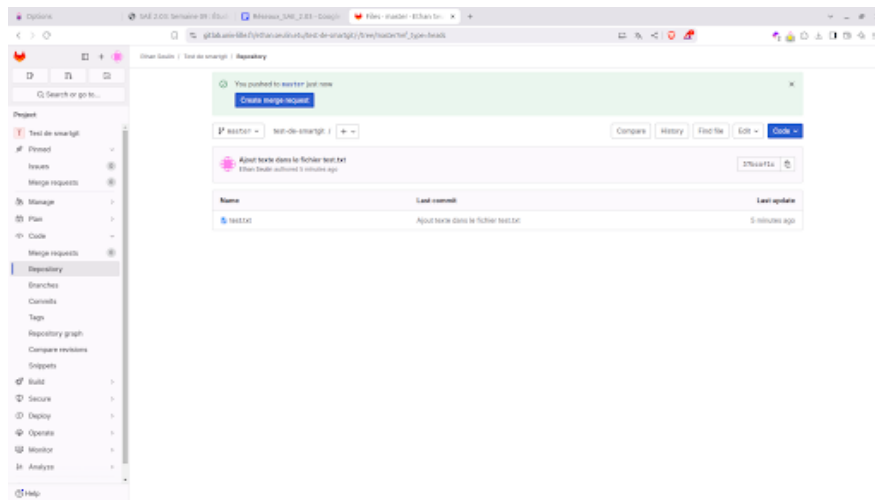


Figure 8: connexion

Comment installer ?

‘sudo apt install gitk’

- Comment le lancer ?

Dans un dépôt Git, exécutez : ‘gitk’

3.1.1 Qu’est-ce que git-gui ?

git-gui est une **interface graphique pour gérer les commits, les branches et les pushes**.

Il est plus orienté **gestion de commits et staging** que gitk, qui est surtout un visualiseur d’historique.

Comment installer ?

sudo apt install git-gui

- Comment le lancer ?

Dans un dépôt Git : git gui

4 Gitea

4.1 QUESTIONS

- Qu'est-ce que Gitea ?

Gitea est un logiciel de gestion de dépôts Git open source qui peut être utilisé pour héberger des projets Git de manière privée ou publique. Il est conçu pour être facile à utiliser et à installer et offre une interface Web pour gérer les dépôts Git.

En plus de ses fonctionnalités de base de gestion de dépôts Git, Gitea offre également des fonctionnalités avancées pour les utilisateurs expérimentés, comme la gestion de tickets et de tâches, le suivi des bugs et la gestion de la documentation.

- À quels logiciels bien connus dans ce domaine peut-on le comparer (en citer au moins 2) ?

Gitea est similaire à d'autres outils de gestion de dépôts **Git** tels que **GitHub** et **GitLab**, mais il est conçu pour être plus léger et plus facile à utiliser.

- Qu'est-ce qu'un fork (dans le domaine du développement logiciel bien entendu) ?

Un fork, c'est un **nouveau logiciel** créé à partir du code source d'un logiciel existant.

- De quel logiciel Gitea est-il le fork ? Ce logiciel existe-t-il encore ?

Gitea est un fork de **Gogs**. Un outil pour gérer des dépôts Git via une interface web. Cela propose une gestion des tickets attachés au projet, des pages de Wiki, une gestion des utilisateurs et des groupes d'utilisateurs et des liens vers de l'intégration continue.

Aujourd'hui, Gogs existe toujours et est utilisé par certaines entreprises tel que steelants et keenton.

4.2 Installation gitea:

4.2.1 Port-forwarding sur VM:

Gitea utilise par défaut le port 3000 pour communiquer, pour cela il faut aller dans les paramètres réseaux de la VM et sélectionner comme type de réseau: "NAT" et faire une règle "gitea" redirigeant les entrées et sorties de la VM sur le port 3000

4.2.2 Pré-requis

Tout d'abord mettre à jour les paquets existants et installer les paquets nécessaires: `sudo apt update && sudo apt upgrade && sudo apt install \-y git sqlite3 curl`

4.2.3 Téléchargements

Télécharger le fichier d'installation Gitea:

```
wget -O gitea https://dl.gitea.com/gitea/main-nightly/gitea-main-nightly-linux-amd64
```

Après avoir été installé,, toujours dans le même répertoire exécutez cette commande pour rendre le fichier exécutable:

```
chmod +x gitea
```

4.2.4 Vérifications des fichiers téléchargés avec GPG

Téléchargez le fichier de signature correspondant à la version de Gitea téléchargée (soit 1.23.5) avec la commande suivante :

```
wget -O gitea-asc https://dl.gitea.com/gitea/1.23.5/gitea-1.23.5-linux-amd64.asc
```

Puis récupérez la clef pour faire la comparaison:

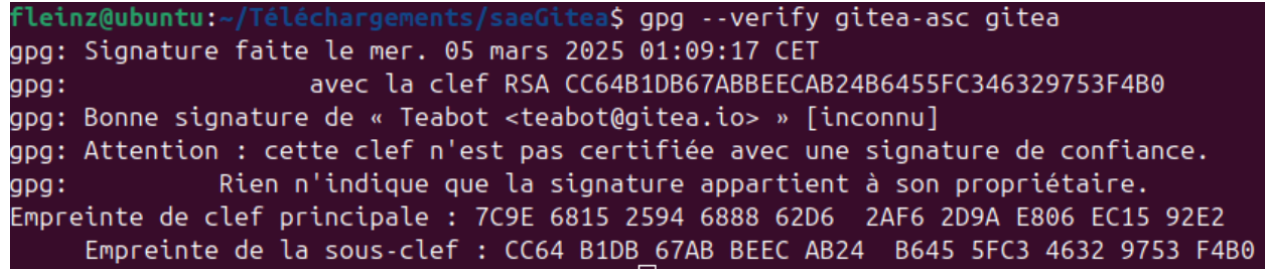
```
gpg --keyserver keys.openpgp.org --recv 7C9E68152594688862D62AF62D9AE806EC1592E2
```

Vérifiez la signature du fichier:

```
gpg --verify gitea-asc gitea
```

Lorsque vous vérifiez la signature GPG d'un fichier binaire, vous pouvez rencontrer un avertissement semblable à celui-ci : `gpg: Attention: cette clef n'est pas certifiée avec une signature de confiance. gpg: Rien n'indique que la signature appartient à son propriétaire. Malgré cet avertissement, tant que le message indique une "Bonne signature", cela confirme que le fichier gitea n'a pas été altéré, et vous pouvez considérer le binaire comme authentique et sûr à utiliser.**`

Puis vérifier la signature avec: `"gpg --verify gitea-asc gitea"`



```
Fleinz@ubuntu:~/Téléchargements/saeGitea$ gpg --verify gitea-asc gitea
gpg: Signature faite le mer. 05 mars 2025 01:09:17 CET
gpg:                avec la clef RSA CC64B1DB67ABBEECAB24B6455FC346329753F4B0
gpg: Bonne signature de « Teabot <teabot@gitea.io> » [inconnu]
gpg: Attention : cette clef n'est pas certifiée avec une signature de confiance.
gpg:                Rien n'indique que la signature appartient à son propriétaire.
Empreinte de clef principale : 7C9E 6815 2594 6888 62D6  2AF6 2D9A E806 EC15 92E2
Empreinte de la sous-clef : CC64 B1DB 67AB BEEC AB24  B645 5FC3 4632 9753 F4B0
```

Figure 9: Console gitea

À l'aide de `'git -v'` j'ai la version, elle est supérieure à la 2.0.

4.3 Préparer l'environnement Gitea

4.3.1 Version Git

Tout d'abord vérifiez si votre version git est bien supérieur à 2.0 avec `git -v`

4.3.2 Utilisateur Git

Ne pas oublier d'avoir les droits sudo.

Pour ajouter l'utilisateur Gitea:

```
sudo adduser \
--system \
--shell /bin/bash \
--gecos 'Git Version Control' \
--group \
--disabled-password \
--home /home/git \
```

Pour créer les répertoires demandés:

```
sudo mkdir -p /var/lib/gitea/{custom,data,log}
sudo chown -R git:git /var/lib/gitea/
sudo chmod -R 750 /var/lib/gitea/
sudo mkdir /etc/gitea
sudo chown root:git /etc/gitea
sudo chmod 770 /etc/gitea
```

Déplacez le fichier gitea dans le répertoire créé, mettez le dans le dossier `/var/lib/gitea`.

4.4 Lancer Gitea

Connectez vous au compte git: `sudo su - git`

Allez dans le dossier d'installation gitea: `cd /var/lib/gitea`

Lancez Gitea: `./gitea web`

4.4.1 Configurer Gitea

4.4.1.1 Paramètres de la base de données

Paramètre	Valeur
Type de base de données	SQLite3
Emplacement	/var/lib/gitea/data/gitea.db

4.4.1.2 Configuration générale

Paramètre	Valeur
Titre du site	votre nom de serveur
Emplacement racine des dépôts	/var/lib/gitea/data/gitea-repositories
Répertoire racine Git LFS	/var/lib/gitea/data/lfs
Exécuter avec le compte d'un autre utilisateur	git
Domaine du serveur	localhost
Port du serveur SSH	22
Port d'écoute HTTP de Gitea	3000
URL de base de Gitea	http://localhost:3000/
Chemin des journaux	/var/lib/gitea/log
Vérification automatique des mises à jour	Cochez cette option

4.4.1.3 Paramètres facultatifs

Paramètre	Valeur
Nom d'utilisateur administrateur	gitea
Courriel	git@localhost
Mot de passe	gitea

4.4.1.4 Fin de l'installation Gitea Cliquez sur installer et suivez l'avancement dans votre console. Le serveur Gitea est maintenant installé et opérationnel. La gestion de certains paramètres peuvent se faire depuis ce lien: <http://localhost:3000/-/admin>

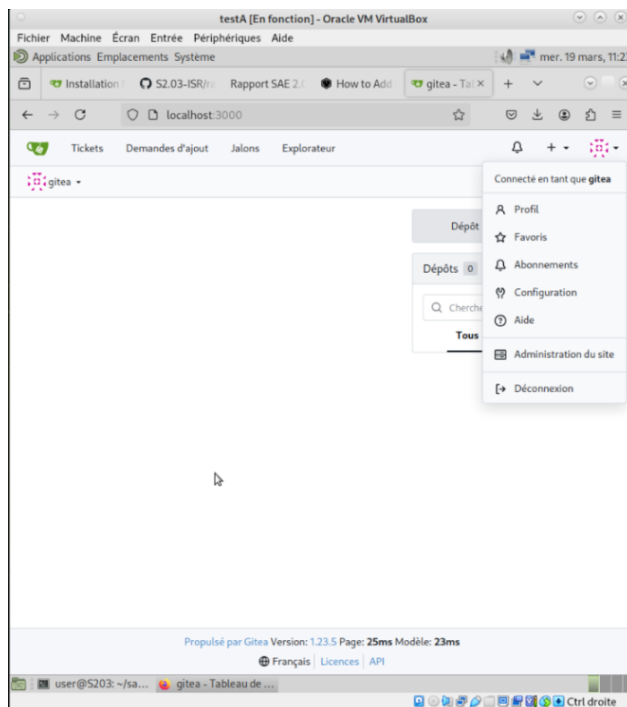
4.4.2 Protéger les fichiers Gitea

De retour avec votre compte changez l'owner du fichier avec: `chown git /var/lib/gitea`
Puis `sudo chmod 750 /etc/gitea` et `sudo chmod 640 /var/lib/gitea/custom/conf/app.ini`

4.5 Lancer Gitea après installation

Pour lancer gitea il vous suffit de vous connecter à l'utilisateur git

Le service gitea maintenant installée sur notre machine, on peut voir que nous sommes connectés à notre compte admin gitea:



4.6 Version Gitea:

Pour vérifier la version de gitea on peut faire la commande:

- `./gitea -v` (identique à git)

4.7 Installer une version spécifique de Gitea:

Par exemple pour télécharger la version 1.24.3 de Gitea il suffit de:

```
wget -O gitea-1.24.3 https://dl.gitea.com/gitea/1.22.0/gitea-1.24.3-linux-amd64
```

Depuis le dossier `/var/lib/gitea` depuis le compte `git` puis d'exécuter `./gitea-1.24.3 web`

4.8 Utilisation de Gitea:

Pour créer un répertoire depuis l'interface web de gitea:

Depuis ma machine principale je me suis connecté au compte admin gitea:

J'ai créé un compte utilisateur depuis l'interface web admin:

Je peux m'y connecter.

Ici j'ai d'abord ajouté mon compte "ethanseulin" comme collaborateur depuis le compte qui a créé le dépôt. Ensuite j'ai cloné mon répertoire depuis l'adresse local du serveur gitea "<http://localhost:3000/gitea/rendu-SAE-ISR>":

- `git clone http://localhost:3000/gitea/rendu-SAE-ISR`

Puis finalement j'ai commit et push les nouveaux fichiers et tout marche à merveille.

Toute cette procédure a été réalisée depuis le pc et non la machine virtuelle, donc le port-forwarding fonctionne bien.

On peut voir du coup les modifications apportés depuis le compte "ethanseulin" sur le dépôt.

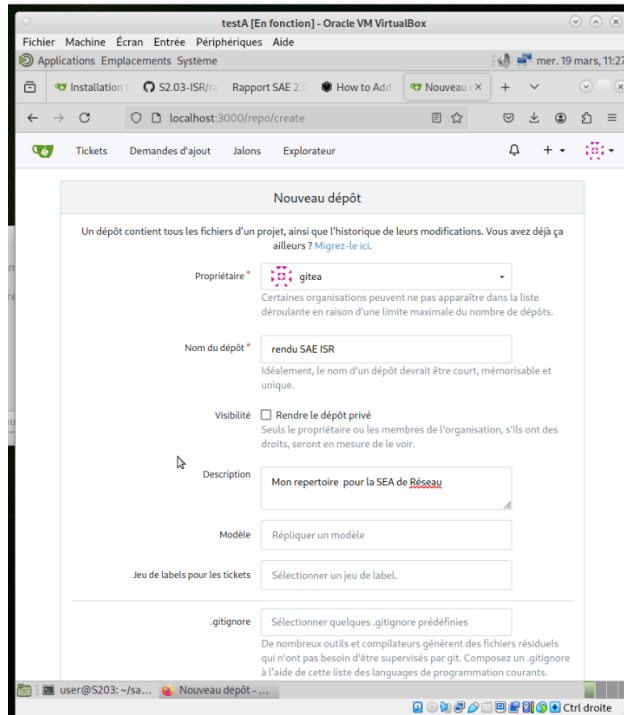


Figure 10: Répertoire création

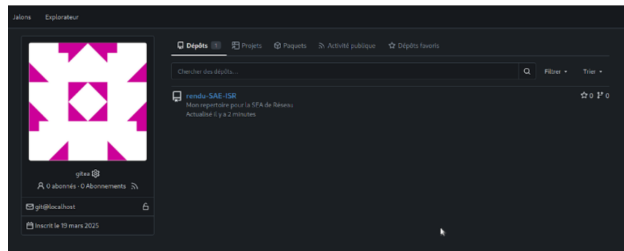


Figure 11: Connexion compte admin

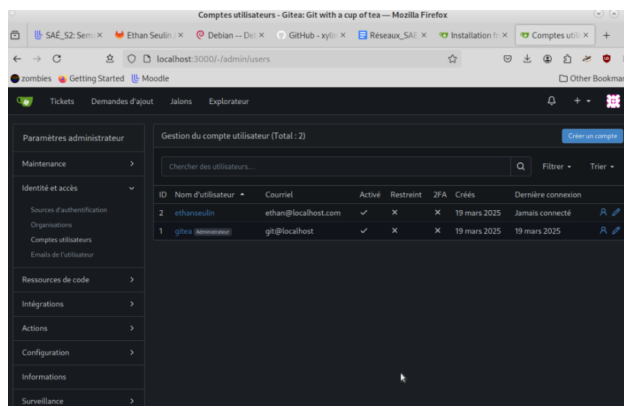


Figure 12: Compte utilisateur

```

Terminal:3
ethan.seulin.etu@boulleau24:~/Téléchargements/sae-r2.03/rendu-SAE-ISR [147]
$ls
imgs MAIN.html README.md
ethan.seulin.etu@boulleau24:~/Téléchargements/sae-r2.03/rendu-SAE-ISR [0]
$git add *
ethan.seulin.etu@boulleau24:~/Téléchargements/sae-r2.03/rendu-SAE-ISR [0]
$git commit -m "init"
[main (root-commit) b55e0f4] init
14 files changed, 641 insertions(+)
create mode 100644 MAIN.html
create mode 100644 README.md
create mode 100644 imgs/SmartGit.png
create mode 100644 imgs/SmartGit2.png
create mode 100644 imgs/accueilDebian.png
create mode 100644 imgs/additionsGuest.png
create mode 100644 imgs/ajoutFichier.png
create mode 100644 imgs/ajoutFichier2.png
create mode 100644 imgs/ajoutServeur.png
create mode 100644 imgs/commandeSudo.png
create mode 100644 imgs/connexion.png
create mode 100644 imgs/logiciels.png
create mode 100644 imgs/miroir.png
create mode 100644 imgs/voirHistorique.png
ethan.seulin.etu@boulleau24:~/Téléchargements/sae-r2.03/rendu-SAE-ISR [0]
$git push
Username for 'http://localhost:3000': ethanseulin
Password for 'http://ethanseulin@localhost:3000':
Enumerating objects: 17, done.
Counting objects: 100% (17/17), done.
Delta compression using up to 16 threads
Compressing objects: 100% (17/17), done.
Writing objects: 100% (17/17), 709.10 KiB | 30.83 MiB/s, done.
Total 17 (delta 0), reused 0 (delta 0), pack-reused 0
remote: . Processing 1 references
remote: Processed 1 references in total
To http://localhost:3000/gitea/rendu-SAE-ISR
 * [new branch]    main -> main
ethan.seulin.etu@boulleau24:~/Téléchargements/sae-r2.03/rendu-SAE-ISR [0]

```

Figure 13: Connexion

