
Rapport SAE 2.03 ISR

Table des matières

| | |
|--|----|
| Semaine 1: Installation et Configuration de Debian..... | 2 |
| 1.1: Réponses aux questions..... | 2 |
| 1.2: Rapport sur nos actions réalisées durant cette semaine..... | 6 |
| 1.3: Procédure pour l'Auto-Installation de Debian | 11 |
| Semaine 2: Apprentissage Balisage Léger - Base | 15 |
| Semaine 3: Recherche et étude des applications clientes..... | 16 |
| 3.1: Réponses aux questions..... | 16 |
| 3.2: Rapport sur nos actions réalisées durant cette semaine..... | 17 |
| Semaine 4: Gitea : Installation & Utilisation du Service..... | 32 |
| 4.1: Réponses aux questions..... | 32 |
| 4.2: Rapport sur nos actions réalisées durant cette semaine..... | 34 |
| 4.3: Procédure pour l'installation de Gitea | 58 |

Résumé

Ce rapport documente les étapes et apprentissages réalisés durant la SAE2.03 ISR.

Il est structuré en plusieurs semaines de travail. La première concerne l'installation et la configuration de Debian, avec la mise en place d'une machine virtuelle et l'automatisation de son installation. Ensuite, l'apprentissage du balisage léger qui n'est pas rédigée. La troisième semaine est dédiée à la recherche et à l'étude des applications clientes Git, comparant différentes interfaces comme GitK, Git-Gui et Git Cola. Enfin, la dernière semaine traite de l'installation et de l'utilisation de Gitea, un service d'hébergement Git, en expliquant son installation, ses paramétrages et sa configuration en tant que service.

Semaine 1: Installation et Configuration de Debian.

Durant la première semaine de la SAE, l'installation de la machine virtuelle ainsi qu'une grande partie des tâches demandées ont été réalisées sous Windows. Nous avons procédé légèrement différemment, mais nous avons tout de même compris le concept.

1.1: Réponses aux questions.

Partie 2.2: Préparation de la Machine Virtuelle

- Que signifie “64-bit” dans “Debian 64-bit” ?

“64-bit” dans “Debian 64-bit” fait référence à la longueur des registres de données et d'adresses utilisés par le processeur de l'ordinateur.

En d'autres termes, un processeur 64 bits peut traiter des données et des instructions en blocs de 64 bits, ce qui lui permet de manipuler des quantités plus importantes d'informations par cycle d'horloge par rapport à un processeur 32 bits.

- Quelle est la configuration réseau utilisée par défaut ?

La configuration réseau utilisée par défaut est le **DHCP** (*Dynamic Host Configuration Protocol*) qui consiste à un périphérique d'obtenir automatiquement une adresse IP, un masque de sous-réseau, une passerelle et des serveurs DNS depuis un routeur ou un serveur DHCP^[1].

- Quel est le nom du fichier XML contenant la configuration de votre machine ?

Le fichier **XML** contenant la configuration de notre machine est nommé **Debian-SAE203.vbox** et suit généralement le format **[nom de la machine].vbox**.

- Sauriez-vous le modifier directement ce fichier de configuration pour mettre 2 processeurs à votre machine ?

Avec nos connaissances actuelles, et en toute honnêteté, non.

Partie 2.3: Installation de l'OS

- Qu'est-ce qu'un fichier iso bootable ?

Un fichier ISO bootable est un fichier permettant à un ordinateur de démarrer directement à travers ce fichier afin d'installer un système d'exploitation sur le disque dur d'un ordinateur.

En effet, il contient les éléments nécessaires au démarrage d'un ordinateur.

- **Qu'est-ce que MATE ? GNOME ?**

MATE et GNOME sont des environnements de bureau (*desktop environment*) qui sont des logiciels permettant d'utiliser un ordinateur à travers un interface graphique utilisateur (*GUI*).

- MATE est un dérivé de GNOME 2 qui s'appuie sur des concepts traditionnels pour GNU/Linux et d'autres systèmes d'exploitation de type Unix^[2]. Il a été créé pour offrir une alternative aux utilisateurs qui préféraient l'expérience classique de GNOME 2, après que GNOME soit passé à la version 3 avec une interface plus moderne et différente.
- GNOME (*GNU Network Object Model Environment*) est l'un des environnements de bureau les plus populaires pour Linux et UNIX. En effet, elle est utilisé par défaut dans Red Hat Enterprise Linux, Ubuntu, Debian, Fedora, openSUSE, Vanilla OS, Endless OS, etc.^[3]

- **Qu'est-ce qu'un serveur web ?**

Un serveur web peut désigner soit un logiciel assurant le service de ressources web (serveur HTTP), soit un ordinateur dédié à répondre aux requêtes du World Wide Web sur un réseau public (Internet) ou privé (intranet), en utilisant principalement le protocole HTTP^[4]. Selon sa nature, il stocke, traite et distribue les fichiers des sites web aux navigateurs des utilisateurs.

- **Qu'est-ce qu'un serveur ssh ?**

Le SSH (*Secure Shell*) désigne à la fois un protocole de communication et un programme informatique. Il permet la connexion d'une machine distante (*serveur*) via une liaison sécurisée dans le but de transférer des fichiers ou des commandes en toute sécurité^[5].

- **Qu'est-ce qu'un serveur mandataire ?**

Un serveur mandataire (*ou « proxy »*) est un logiciel intermédiaire qui s'interpose entre deux hôtes afin de faciliter, contrôler ou sécuriser leurs échanges^[6].

Partie 3.1: Accès sudo pour user

- **Comment peut-on savoir à quels groupes appartient l'utilisateur user ?**

On peut connaître les groupes auxquels appartient l'utilisateur **user** grâce à la

commande :

```
groups user
```

Partie 3.2: Installation des Suppléments Invités

- **Quel est la version du noyau Linux utilisé par votre VM ?**

La version du noyau Linux (*kernel*) utilisé par ma VM est **Linux 6.1.0-31-amd64**. On peut le savoir en utilisant la commande^[7]:

```
uname -r
```

- **À quoi servent les suppléments invités ? Donner 2 principales raisons de les installer.**

Les suppléments invités apportent de nouvelles fonctionnalités entre la machine hôte et invitée^[8]:

- **Amélioration de l'intégration de la souris et de l'affichage** entre la machine hôte et invitée.
- **Dossiers partagés** entre la machine hôte et invitée.
- **À quoi sert la commande mount (dans notre cas de figure et dans le cas général) ?**
- **Dans notre cas:** la commande **mount** est utilisée pour monter le CD contenant les suppléments invités de VirtualBox, cela permet d'exécuter ensuite l'installation des additions invitées avec : **sudo /mnt/VBoxLinuxAdditions.run**
- **Dans le cas général:** **mount** sert à attacher des périphériques de stockage (*clés USB, disques durs, partitions, fichiers ISO, partages réseau, etc.*) à l'arborescence du système, afin d'accéder à son contenu^[9].

Partie 4.2: Quelques Questions

- **1. Qu'est-ce que le Projet Debian ? D'où vient le nom Debian ?**

Le "Projet Debian" est une organisation composé de bénévoles et qui a pour but de développer le logiciel libre et promouvoir les idéaux de la communauté du logiciel libre^[10]. Concernant l'origine du nom Debian, elle est la contraction des noms Debra

et Ian Murdock, qui a fondé le projet ^[11].

- **2. Quelle sont les durées de ces prises en charge ?**

- La durée minimale est de 3 ans ^[12].
- La durée du LTS est de au moins 5 ans ^[13].
- La durée du eLTS est de 10 ans ^[14].

- **3. Pendant combien de temps les mises à jour de sécurité seront-elles fournies ?**

Les mises à jour de sécurité seront fournis pendant 3 ans après sa publication ^[15].

- **4. Combien de version au minimum sont activement maintenues par Debian ?**

Donnez leur nom générique.

Debian a toujours, au minimum, 3 versions activement maintenues : "stable", "testing" et "unstable" ^[16].

- **5. D'où viennent les noms de code donné aux distributions ?**

Les noms de code donné aux distributions proviennent des personnages de films "Toy Story" par Pixar ^[17]

- **6. Combien et lesquelles sont prises en charge par la version Bullseye ?**

Le nombre d'architecture prise en charge par la version Bullseye est de 9: "amd64", "i386", "ppc64el", "s390x", "armel", "armhf", "arm64", "mipsel" et "mips64el" ^[18].

La source des questions suivantes est la suivante: [Debian Version](#).

- **7. Première version avec un nom de code**

- **Quelle a été le premier nom de code utilisé ?**

Le premier nom de code utilisé est Buzz.

- **Quand a-t-il été annoncé ?**

Elle a été annoncée le 17 juin 1996.

- **Quelle était le numéro de version de cette distribution ?**

Le numéro de version de cette distribution est le 1.1.

- **8. Dernière nom de code attribué**

- **Quel est le dernier nom de code annoncée à ce jour ?**

Le dernier nom de code annoncé à ce jour est Trixie.

- **Quand a-t-il été annoncé ?**
Elle a été annoncée en août 2024.
- **Quelle est la version de cette distribution ?**
Le numéro de version de cette distribution est le 13.

Partie 5.1: Récupérer et préparer les fichiers nécessaires



Cette section détaillera tous les ajouts et modifications que j'ai apportés au fichier `preseed-fr.cfg` afin de répondre aux exigences demandées.

- **Ajouter le droit d'utiliser sudo à l'utilisateur standard**

En me basant sur le guide de préconfiguration Debian concernant la **configuration des comptes**, j'ai ajouté le groupe **sudo** dans `d-i passwd/user-default-groups`:

```
## Utilisateur standard (~ ligne 50)
[...]
d-i passwd/user-default-groups string audio cdrom video sudo
```

- **Installer l'environnement MATE.**

En me basant sur le guide de préconfiguration Debian concernant le **choix des paquets**, j'ai ajouté **mate-desktop** dans `tasksel tasksel/first`:

```
## Installation meta-paquettages (~ ligne 82)
[...]
tasksel tasksel/first multiselect standard, ssh-server, mate-desktop
```

- **Ajouter les paquets suivants : sudo, git, sqlite3, curl, bash-completion, neofetch**

En me basant sur le guide de préconfiguration Debian concernant le **choix des paquets**, j'ai ajouté les lignes suivants:

```
# Paquets supplémentaires (~ ligne 67)
d-i pkgsel/include string sudo git sqlite3 curl bash-completion neofetch
```

1.2: Rapport sur nos actions réalisées durant cette semaine.



Concernant le rapport de nos actions sur la **partie 2 à 4**, nous n'avons pas jugé pertinent de le détailler, car nous nous sommes simplement

conformés aux exigences demandées.

L'automatisation de l'installation d'un système d'exploitation comme Debian simplifie et optimise le déploiement, tout en réduisant les risques d'erreurs humaines. De plus, cela permet de se détendre et de jouer à Minecraft pendant que l'installation se fait, sans avoir à intervenir.

Cette section présente les étapes et les choix techniques qui ont permis de mettre en place un processus d'installation automatisée pour Debian sur Windows.

Pour des raisons qui me sont encore sombre, l'application VirtualBox de ma session dans les salles de TP refuse obstinément de s'ouvrir.

Tristesse et désespoir.

Après avoir créé la machine virtuelle sur VirtualBox dans un répertoire facilement accessible et où OneDrive ne viendrait pas me ralentir, c'est-à-dire dans le dossier "**Downloads**", j'ai téléchargé l'archive **autoinstall_Debian.zip** depuis Moodle et l'ai décompressée dans le répertoire de ma machine virtuelle : < **.\\Downloads\\SAE-203\\Debian-SAE203** >.

Comme demandé, je suis allé dans WSL et j'ai utilisé la commande suivante :

```
cd /mnt/c/Users/ridhi/Downloads/SAE-203/Debian-SAE203/autoinstall_iut_salles_TP && sed -i  
-E "s/(--iprt-iso-maker-file-marker-bourne-sh).*$/\1=$(cat /proc/sys/kernel/random/uuid)"/"  
S203-Debian12.viso
```

Cela m'a permis de me déplacer dans le répertoire où se trouvait le fichier **S203-Debian12.viso** et de remplacer la chaîne "@@UUID@@" par un identifiant unique universel.

J'ai ensuite vérifié que tout était OK en ouvrant le fichier et, effectivement, le "@@UUID@@" avait bien été remplacé par une série de lettres et de chiffres.

J'ai ensuite inséré le fichier **S203_Debian12.viso** dans le lecteur optique de ma machine virtuelle et lancé celle-ci. Et là, c'est le drame : **LES PROBLÈMES COMMENCENT !**

VirtualBox m'affiche une erreur comme quoi il ne trouve pas le fichier **debian[...].iso** :

```
VISO: RTVfsChainOpenFile  
failed to open '/home/  
public/r205/debian-12.9.0-  
amd64-netinst.iso':  
VERR_PATH_NOT_FOUND  
(VERR_PATH_NOT_FOUND).
```

Heureusement, j'avais pris soin de vérifier le fichier **S203-Debian12.viso**, où j'avais remarqué le chemin indiqué dans l'[erreur ci-dessus](#), mais auquel je n'avais pas prêté attention:

```
--iprt-iso-maker-file-marker-bourne-sh=d6246a32-496a-40f2-9f18-026d21cec1e0  
--volume-id=S203-viso --file-mode=0444 --dir-mode=0555 --no-file-mode --no-dir-mode  
--import-iso=/home/public/r205/debian-12.9.0-amd64-netinst.iso --file-mode=0444 --dir-mode=0555  
isolinux/isolinux.cfg=:must-remove: 'isolinux/isolinux.cfg=isolinux-isolinux.cfg'  
/isolinux/txt.cfg=:must-remove: '/isolinux/txt.cfg=isolinux-txtr-fr.cfg'  
'/preseed.cfg=preseed-fr.cfg'  
'/vboxpostinstall.sh=vboxpostinstall.sh' --push-iso=/usr/share/virtualbox/VBoxGuestAdditions.iso  
/vboxadditions=/  
--pop
```

J'ai donc téléchargé le fichier **debian[...].iso** depuis leur [site officiel](#) et je l'ai placé dans le répertoire < .\Downloads\SAE-203\Debian-SAE203\autoinstall_iut_salles_TP\ressources >.

J'ai ensuite modifié le fichier **S203-Debian12.viso** pour y insérer un chemin absolu vers ce fichier **debian[...].iso** en écrivant :

```
C:\Users\ridhi\Downloads\SAE-203\Debian-SAE203\autoinstall_iut_salles_TP\S203-  
Debian12.viso
```

Et ça n'a pas marché.

Du coup, j'ai changé de stratégie, passé en mode Linux (*en écrivant "mon/chemin" au lieu de "mon|chemin"*) et utilisé un chemin relatif, comme le montre l'image ci-dessous.

```
--iprt-iso-maker-file-marker-bourne-sh=ca72a684-2a9e-49a7-b83f-181b7508aed5
--volume-id=S203-viso --file-mode=0444 --dir-mode=0555 --no-file-mode --no-dir-mode
--import-iso=./ressources/debian-12.9.0-amd64-netinst.iso --file-mode=0444 --dir-mode=0555
isolinux/isolinux.cfg=:must-remove: 'isolinux/isolinux.cfg=isolinux-isolinux.cfg'
/isolinux/txt.cfg=:must-remove: '/isolinux/txt.cfg=isolinux-txt-fr.cfg'
'/preseed.cfg=preseed-fr.cfg'
'/vboxpostinstall.sh=vboxpostinstall.sh' --push-iso=./ressources/VBoxGuestAdditions.iso
/vboxadditions=/
--pop
```

J'ai redémarré la machine, mais une nouvelle erreur est apparue, différente de la précédente.

```
VISO: RTVfsChainOpenFile
failed to open '/usr/share/virtualbox/VBoxGuestAdditions.iso':
VERR_PATH_NOT_FOUND
(VERR_PATH_NOT_FOUND).
```

Je suis donc allé chercher sur Internet le fichier **VBoxGuestAdditions.iso**. Je l'ai trouvé sur le site suivant : [VirtualBox Guest Additions Collection : Oracle : Free Download, Borrow, and Streaming : Internet Archive](#).

J'y ai téléchargé la dernière version, modifié son nom et l'ai placé dans le répertoire où se trouvait le fichier **debian[...].iso**, c'est-à-dire dans < .\Downloads\SAE-203\Debian-SAE203\autoinstall_iut_salles_TP\ressources >.

Ensuite, j'ai encore modifié le fichier **S203-Debian12.viso**:

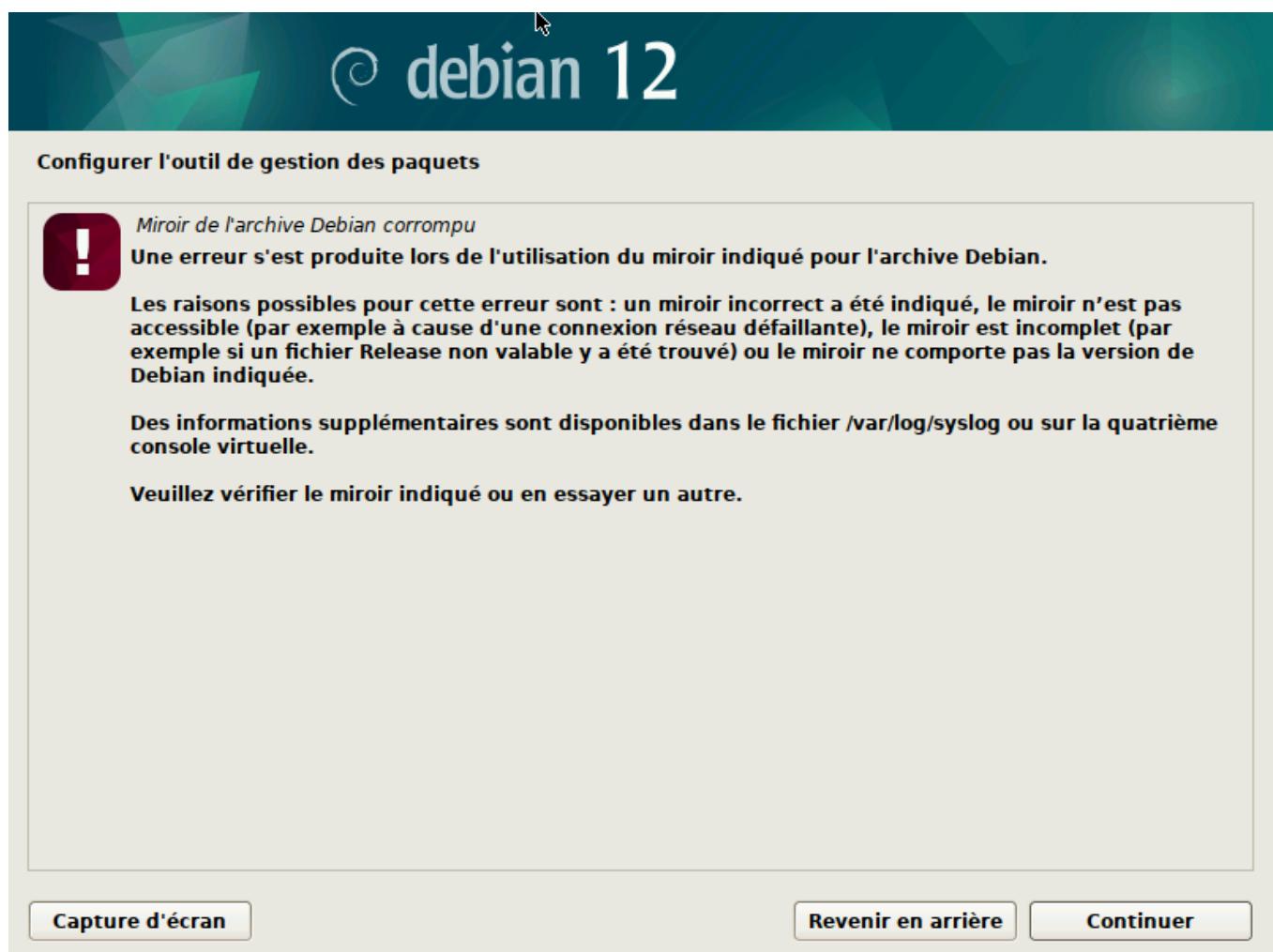
```
--iprt-iso-maker-file-marker-bourne-sh=d6246a32-496a-40f2-9f18-026d21cec1e0
--volume-id=S203-viso --file-mode=0444 --dir-mode=0555 --no-file-mode --no-dir-mode
--import-iso=/home/public/r205/debian-12.9.0-amd64-netinst.iso --file-mode=0444 --dir-mode=0555
isolinux/isolinux.cfg=:must-remove: 'isolinux/isolinux.cfg=isolinux-isolinux.cfg'
/isolinux/txt.cfg=:must-remove: '/isolinux/txt.cfg=isolinux-txt-fr.cfg'
'/preseed.cfg=preseed-fr.cfg'
'/vboxpostinstall.sh=vboxpostinstall.sh' --push-iso=/usr/share/virtualbox/VBoxGuestAdditions.iso
/vboxadditions=/
--pop
```

Fichier S203-Debian12.viso avant

```
--iprt-iso-maker-file-marker-bourne-sh=ca72a684-2a9e-49a7-b83f-181b7508aed5
--volume-id=S203-viso --file-mode=0444 --dir-mode=0555 --no-file-mode --no-dir-mode
--import-iso=../ressources/debian-12.9.0-amd64-netinst.iso --file-mode=0444 --dir-mode=0555
isolinux/isolinux.cfg=:must-remove: 'isolinux/isolinux.cfg=isolinux-isolinux.cfg'
/isolinux/txt.cfg=:must-remove: '/isolinux/txt.cfg=isolinux-txt-fr.cfg'
'/preseed.cfg=preseed-fr.cfg'
'vboxpostinstall.sh=vboxpostinstall.sh' --push-iso=../ressources/VBoxGuestAdditions.iso
/vboxadditions=/
--pop
```

Fichier S203-Debian12.viso après

Enfin, tout semblait fonctionner, et la machine a commencé son installation... jusqu'à ce qu'une nouvelle erreur apparaisse.



Je me suis souvenu de ce que M. Carle avait mentionné dans son PDF, affiché sur mon autre écran, concernant le fichier **preseed.cfg** :

"Pour un usage autre (chez vous ou avec une autre version de Debian), il faudra adapter ce fichier de pré-configuration (par exemple, commenter la ligne définissant le proxy)".

J'ai donc commenté la section concernant le proxy:

```
58
59  ### Packages, Mirrors, Image
60  ## Proxy : Obligatoire à L'université
61  d-i mirror/http/proxy string http://cache.univ-lille.fr:3128
62
```

En recréant la VM pour prendre des captures d'écran pour ce rapport, j'ai remarqué que le Wi-Fi que j'utilisais jouait un rôle par rapport au proxy.



En clair : quand je suis connecté à **eduroam**, tout fonctionne, mais dès que je suis sur le Wi-Fi de ma résidence, ça plante (ce n'est pas comme si c'était écrit en gros sur l'image concernant l'[erreur de proxy](#), mais bon...).

Enfin, tout est devenu fonctionnel et l'installation s'est parfaitement déroulée.

1.3: Procédure pour l'Auto-Installation de Debian

Cette section vous guidera dans l'installation automatique de Debian sur Windows, ainsi que sur Linux et d'autres systèmes (*où vous devez adapter la méthode*).

Prérequis

1. **VirtualBox:** Assurez-vous que VirtualBox est bien installé et fonctionnel avec son «extension pack» sur la machine hôte. Si ce n'est pas le cas, vous pouvez le télécharger et l'installer depuis le site officiel : [VirtualBox](#).
2. **L'archive contenant les fichiers nécessaires à l'installation d'une Debian sur l'application VirtualBox [ici](#).**
3. **Une image ISO de Debian [disponible ici](#).**
4. **Une image ISO VBoxGuestAdditions:** Cette image contient les VirtualBox Guest Additions, qui sont un ensemble de pilotes et d'outils destinés à améliorer les performances et l'intégration du système d'exploitation invité (OS dans une machine virtuelle) avec l'hôte (le système d'exploitation de la machine physique). Vous pouvez le télécharger en cliquant [ici](#).

5. Un éditeur de texte pour pouvoir modifier le fichier **preseed-fr.cfg** et **S203-Debian12.viso**.
6. Assurez-vous d'avoir au moins 25 Go d'espace libre sur votre disque dur.
7. Vous devez avoir au minimum 4 Go de RAM.

Création de la machine virtuel

- Démarrez VirtualBox et créez une nouvelle machine virtuelle en cliquant sur "Nouvelle".
- Dans la section **Name and Operating System**:

| Label | Input |
|-----------|--|
| Nom | Entrez le nom de la machine (un nom significatif ou symbolique). |
| Folder | Choisissez le répertoire où votre machine virtuelle sera enregistrée et stockée. |
| ISO Image | Rien. |
| Edition | Rien. |
| Type | Linux |
| Subtype | Debian |
| Version | Debian 64-bit ou Debian 12 Bookworm |

- Dans la section **Hardware**:

| Label | Input |
|--------------|--------------------------|
| Mémoire vive | 2048MB |
| Processors | 1 |
| Enable EFI | <input type="checkbox"/> |

- Dans la section **Hard Disk**:

| Label | Input |
|--------------------------------|-------------------------------------|
| Create a Virtual Hard Disk now | <input checked="" type="checkbox"/> |

| | |
|----------------------------------|--|
| Hard Disk File Location and Size | Dans le même répertoire que celui mentionné précédemment (<i>Folder</i>), 20 Go d'espace disque suffisent. |
| Hard Disk File Type and Variant | Pas besoin. |

- Cliquez sur "Finish".

Préparation des fichiers nécessaire pour l'installation

- Récupérez l'archive contenant les fichiers nécessaires à l'installation de Debian [ici](#) et décompressez-la dans le répertoire où vous avez créé votre machine virtuelle précédemment.
- Téléchargez l'image [ISO de Debian](#) ainsi que [VirtualBox Guest Additions](#).
- Une fois les fichiers .iso téléchargée, placez le fichier dans le répertoire **ressource** qui se trouve à l'intérieur de l'archive que vous avez décompressé précédemment.
- Ouvrez votre Terminal et entrez la commande suivant:

```
cat /proc/sys/kernel/random/uuid
```

- Copiez le résultat de la commande, puis ouvrez le fichier **S203-Debian12.viso**, qui se trouve dans l'archive décompressée. Remplacez la chaîne **@@UUID@@** par le résultat de la commande en collant ce que vous avez copié.
- Toujours dans le fichier **S203-Debian12.viso**, remplacez la chaîne située à la troisième ligne **{votre fichier debian[...].iso}** par le nom du fichier Debian ISO que vous avez téléchargé.
- Ensuite, à la fin du fichier, remplacez **{votre fichier VBGA.iso}** par le nom de votre fichier **VirtualBox Guest Additions**, qui se trouve dans le répertoire **ressources**. (*Assurez-vous d'ajouter l'extension .iso si elle n'apparaît pas lorsque vous collez le nom du fichier.*)
- Enfin, si nécessaire, et en vous basant sur le [contenu du fichier de préconfiguration pour Bookworm](#), vous pouvez modifier le fichier **preseed-fr.cfg**.

Installation Automatique de la machine virtuelle

Nous arrivons à la fin de la procédure pour l'auto-installation de Debian.

1. Retournez sur VirtualBox.
2. Sélectionnez la machine virtuelle que vous avez créée.
3. Cliquez sur **Configuration**.
4. Allez dans l'onglet **Stockage**.
5. Sélectionnez **Contrôleur: IDE**, puis cliquez sur l'icône de disque avec un "+".
6. Cliquez sur **Ajouter**, puis recherchez et sélectionnez **S203-Debian12.viso** dans votre répertoire.
7. Cliquez sur **Choose**, puis supprimez le disque vide en faisant un clic droit dessus et en sélectionnant **Supprimer**.
8. Cliquez sur **OK**, puis démarrez votre machine. L'installation automatique débutera.
9. Une fois l'installation terminée, voici les identifiants de votre machine :

| Nom Utilisateur | Mot de passe |
|-----------------|--------------|
| root | root |
| user | user |

Semaine 2: Apprentissage Balisage Léger - Base

Semaine 3: Recherche et étude des applications clientes.

3.1: Réponses aux questions.

Partie 1.2: Les interfaces graphiques pour git

- **Qu'est-ce que le logiciel gitk ? Comment se lance-t-il ?**

`gitk` est un logiciel qui affiche l'historique des **commits** d'un dépôt Git. Il affiche les branches, les tags, et les modifications dans une vue arborescente^[19]. Il se lance en exécutant la commande suivante dans un terminal, depuis le répertoire d'un dépôt Git :

```
gitk
```

- **Qu'est-ce que le logiciel git-gui ? Comment se lance-t-il ?**

`git-gui` est une interface graphique permettant de préparer des commits, gérer les index, faire des push, pull et merge^[20]. Il se lance en exécutant la commande suivante dans un terminal, toujours dans le répertoire d'un dépôt Git :

```
git gui
```

Partie 1.3: Installons autre chose et comparons

- **Pourquoi avez-vous choisi ce logiciel ?**

Git Cola est un bon choix, car il est léger, open source, propose une interface claire pour gérer les dépôts Git et me rend nostalgique du *Coca Cola*.

- **Comment l'avez vous installé ?**

Je l'ai installé en suivant les instructions de la page de téléchargement de **Git Cola** puis en exécutant la commande suivante :

```
sudo apt install git-cola
```

- **Comparer-le aux outils inclus avec git ainsi qu'avec ce qui serait fait en ligne de commande pure : fonctionnalités avantage, inconvénients...**

- **Gitk :**
 - Avantages : Simple, rapide, montre un bon historique visuel.
 - Inconvénients : Interface un peu vieillotte, manque de fonctionnalités avancées.
- **Git-Gui :**
 - Avantages : Bonne gestion des index, possibilité de commit visuellement.
 - Inconvénients : Interface minimalist, pas très intuitive pour les débutants.
- **Git-Cola :**
 - Avantages : Interface moderne, nombreuses fonctionnalités, personnalisable.
 - Inconvénients : Peut être plus lourd et complexe à prendre en main.
- **Ligne de commande Git :**
 - Avantages : Puissante, flexible, tout est accessible via des commandes.
 - Inconvénients : Courbe d'apprentissage plus élevée, nécessite de mémoriser des commandes.

3.2: Rapport sur nos actions réalisées durant cette semaine.

Configuration Globale de Git

J'ai commencé par ouvrir ma machine virtuelle et utiliser le raccourci clavier **Ctrl+Alt+T** pour ouvrir le Terminal.

J'ai ensuite suivi les étapes demandées en définissant le nom d'utilisateur global pour Git :

```
git config --global user.name "El-Khair Nourdine"
```

Puis, j'ai défini l'adresse e-mail globale associée aux commits :

```
git config --global user.email "el-khair.nourdine.etu@univ-lille.fr"
```

Enfin, j'ai défini le nom de la branche par défaut lors de l'initialisation d'un nouveau dépôt :

```
git config --global init.defaultBranch "master"
```

Les Interfaces Graphiques pour Git: Initialisation

J'ai commencé à mettre à jour la liste des paquets disponibles en exécutant la commande :

```
sudo apt update && sudo apt upgrade
```

Ensuite, j'ai installé les paquets **gitk** et **git-gui** :

```
sudo apt install gitk && sudo apt install git-gui
```

Avant de les utiliser, j'ai pris le temps de me renseigner sur leur fonctionnement en consultant les sites suivantes :

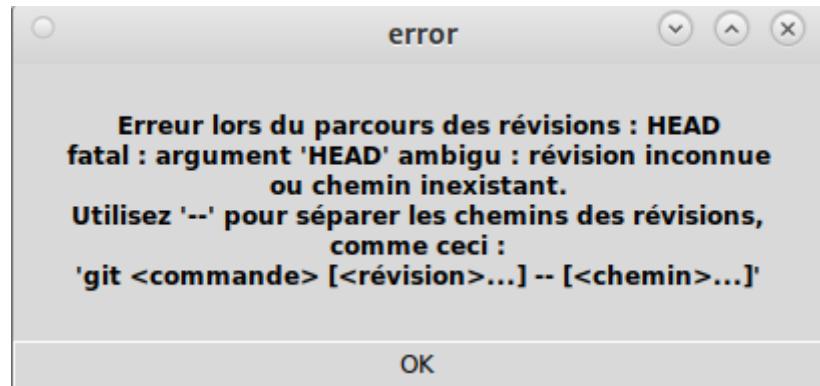
- [Guide Git Gui](#)
- [Guide Gitk](#)

Après avoir compris leur fonctionnement, j'ai transféré l'archive de ce rapport sur le bureau de ma machine virtuelle. J'ai ensuite créé un répertoire sur mon bureau pour y placer mon rapport et commencer à manipuler **gitk** et **git-gui** :

```
mkdir ~/Bureau/git-test && cd ~/Bureau/git-test && mv ../rapport .
```

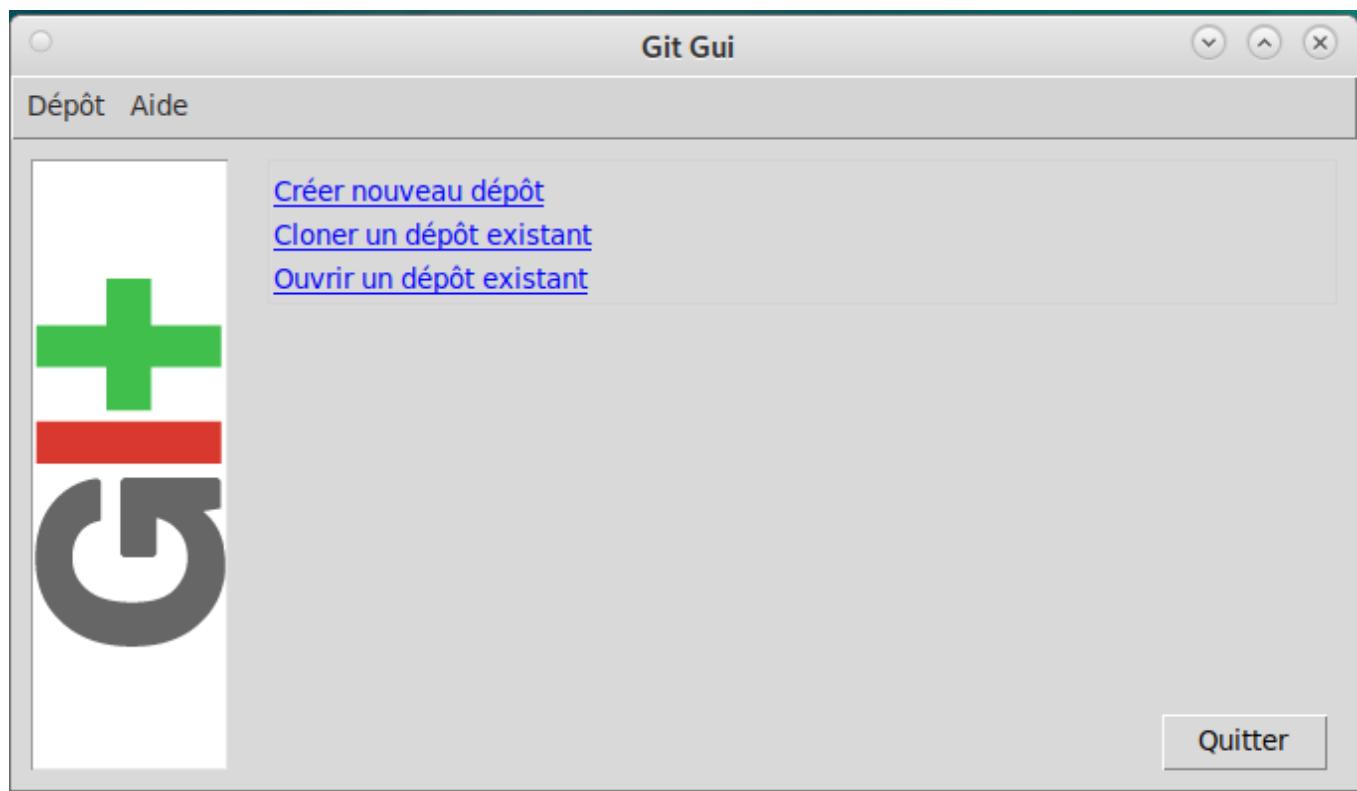
Les Interfaces Graphiques pour Git: Git Gui

J'ai voulu tester **gitk** en exécutant la commande **gitk &** dans le répertoire *git-test*, mais j'ai rencontré un [message d'erreur](#) indiquant que Git ne parvenait pas à interpréter correctement l'argument **HEAD**.



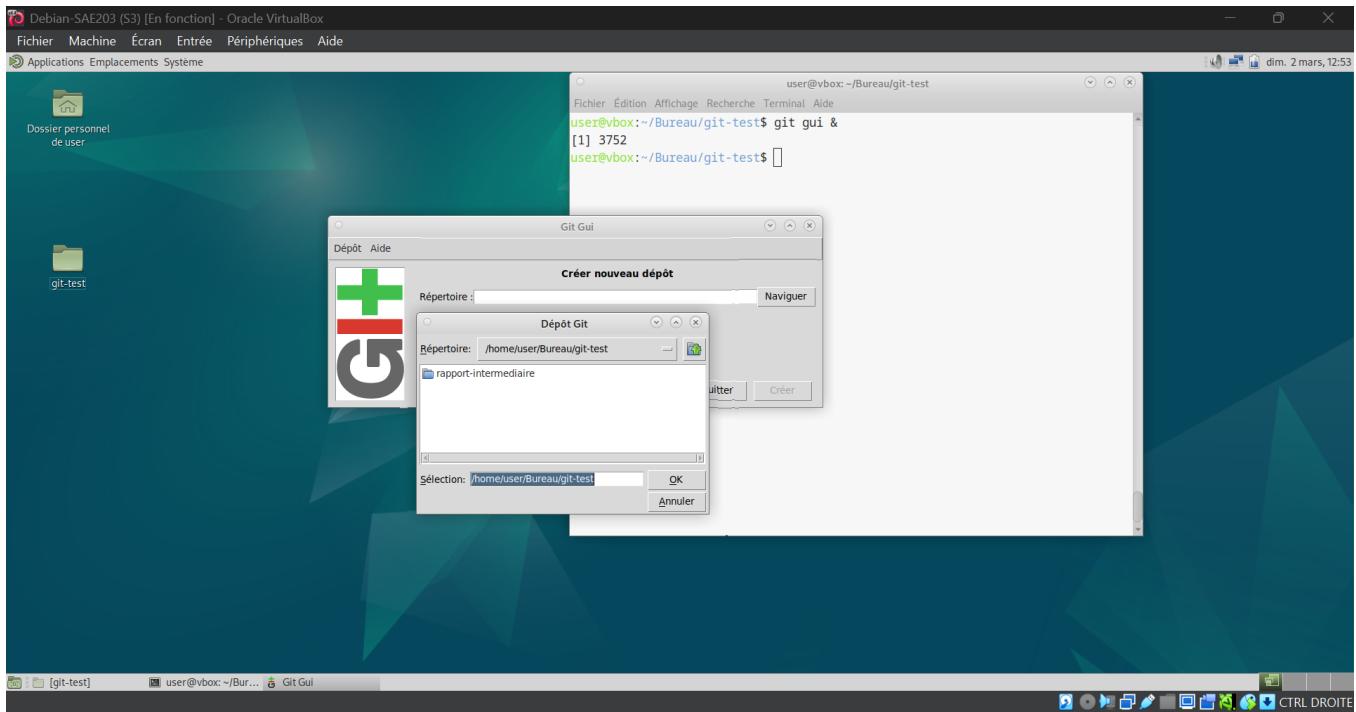
Ce problème venait du fait que je n'avais pas encore initialisé de dépôt Git. J'ai donc commencé par utiliser **git-gui**, en exécutant la commande `git gui &` pour initialiser un dépôt Git avant tester **gitk**.

À l'ouverture, l'interface de Git Gui se présente ainsi :



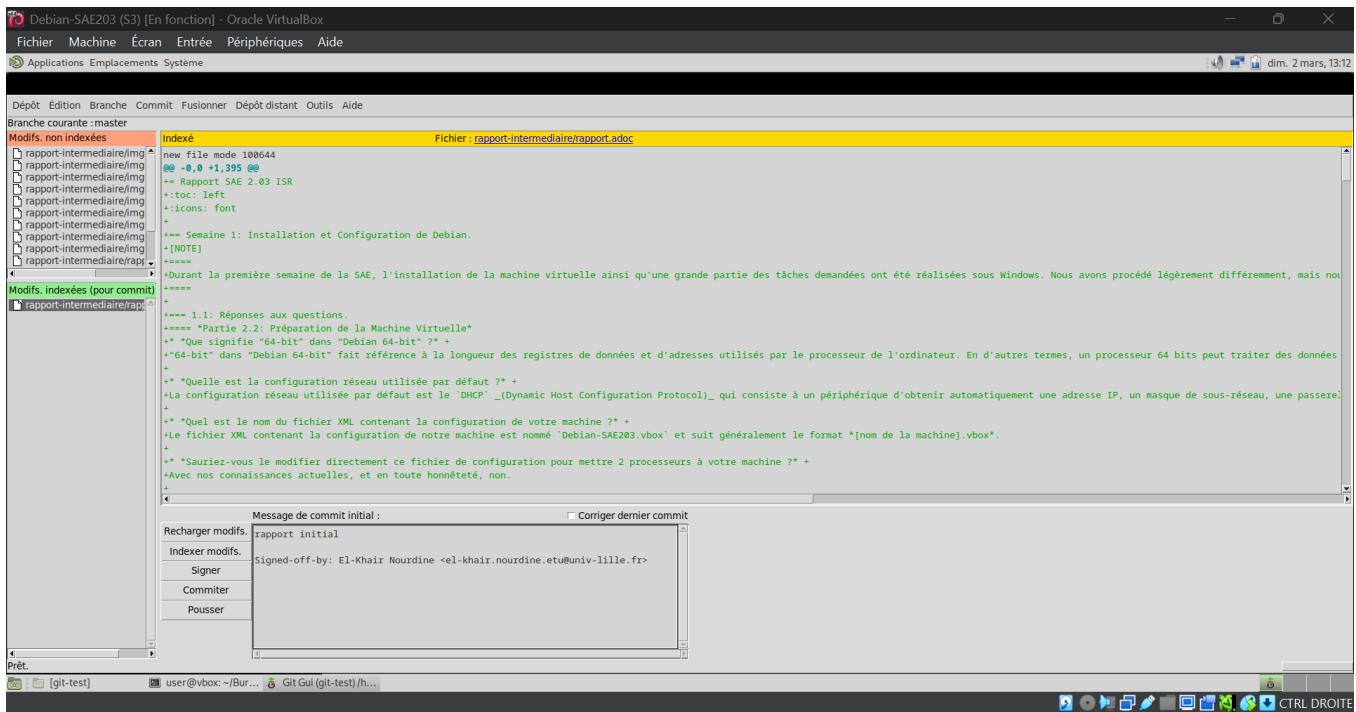
J'ai ensuite créé un nouveau dépôt, ce qui équivaut à la commande `git init`.

Puis, j'ai cliqué sur **Naviguer** et étant donné que j'avais exécuté la commande `git gui &` dans le répertoire contenant mon rapport, je n'ai pas eu à faire beaucoup de manipulations. J'ai simplement cliqué sur OK :



Pour faire un premier essai, j'ai indexé le fichier *rapport.adoc* en cliquant sur l'icône à gauche de son nom. Dans le champ **Message de commit initial**, j'ai saisi : "rapport initial".

J'ai ensuite cliqué sur **Signer**, puis sur **Committer**, ce qui équivaut à la commande : **git commit -m "rapport initial"**

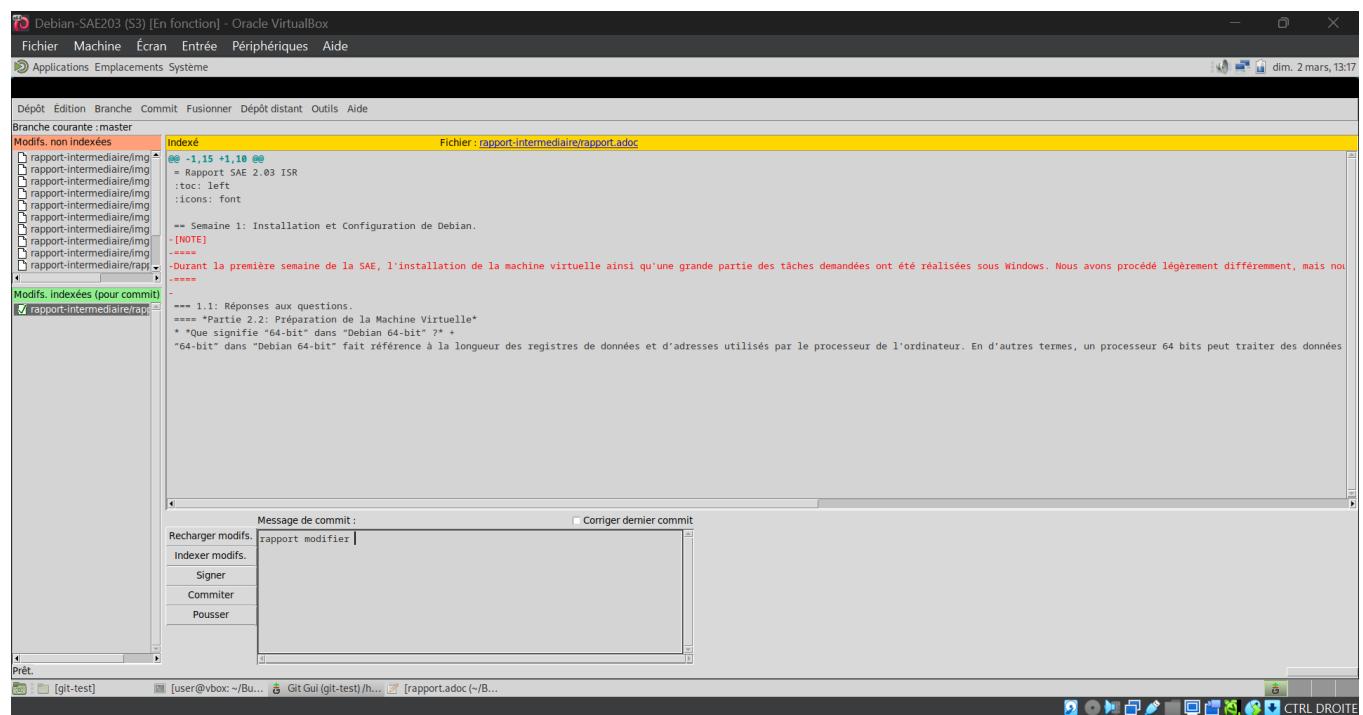


J'ai ensuite modifié le fichier *rapport.adoc* en exécutant la commande suivante dans le terminal :

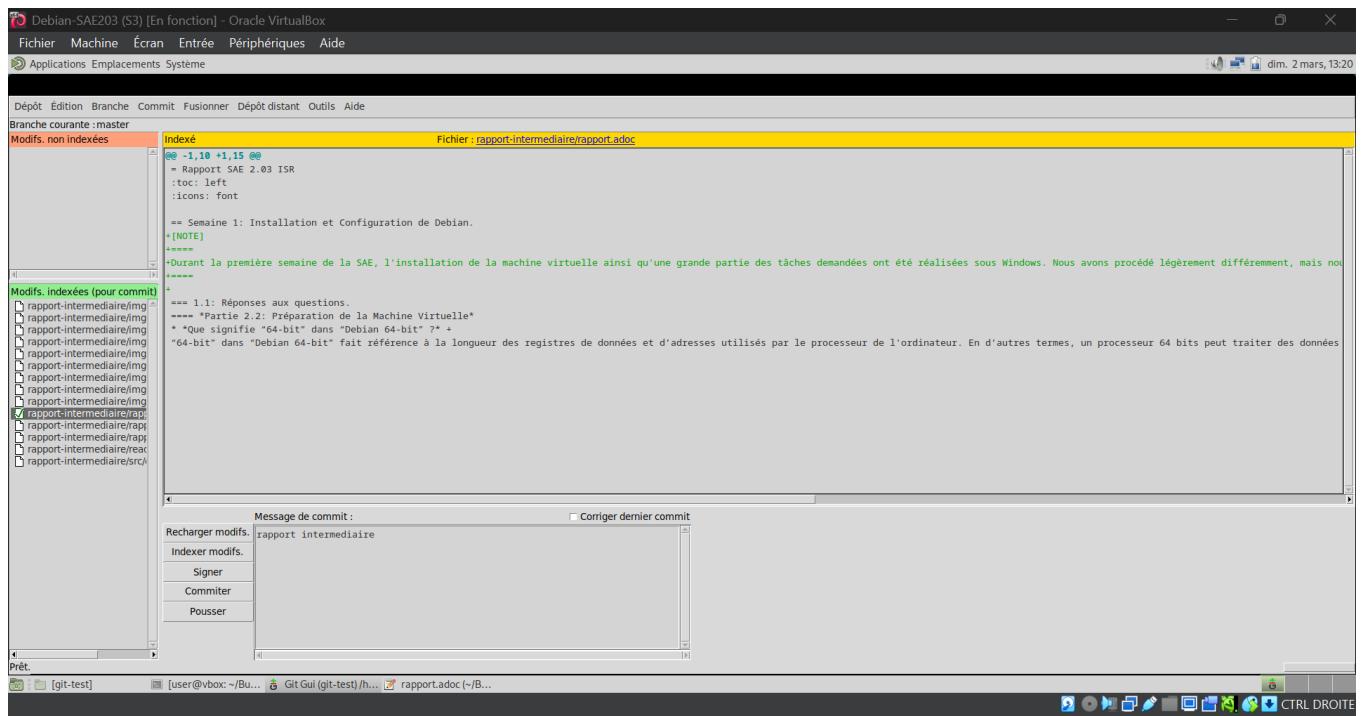
J'ai supprimé une partie du contenu, puis enregistré le fichier.

En revenant dans Git Gui, j'ai cliqué sur **Recharger modifs** pour voir apparaître le *rapport.adoc* qui n'était plus visible à cause du commit précédent.

Ensuite, j'ai cliqué sur l'icône à gauche de son nom pour l'indexer, visualiser les modifications et les committer:



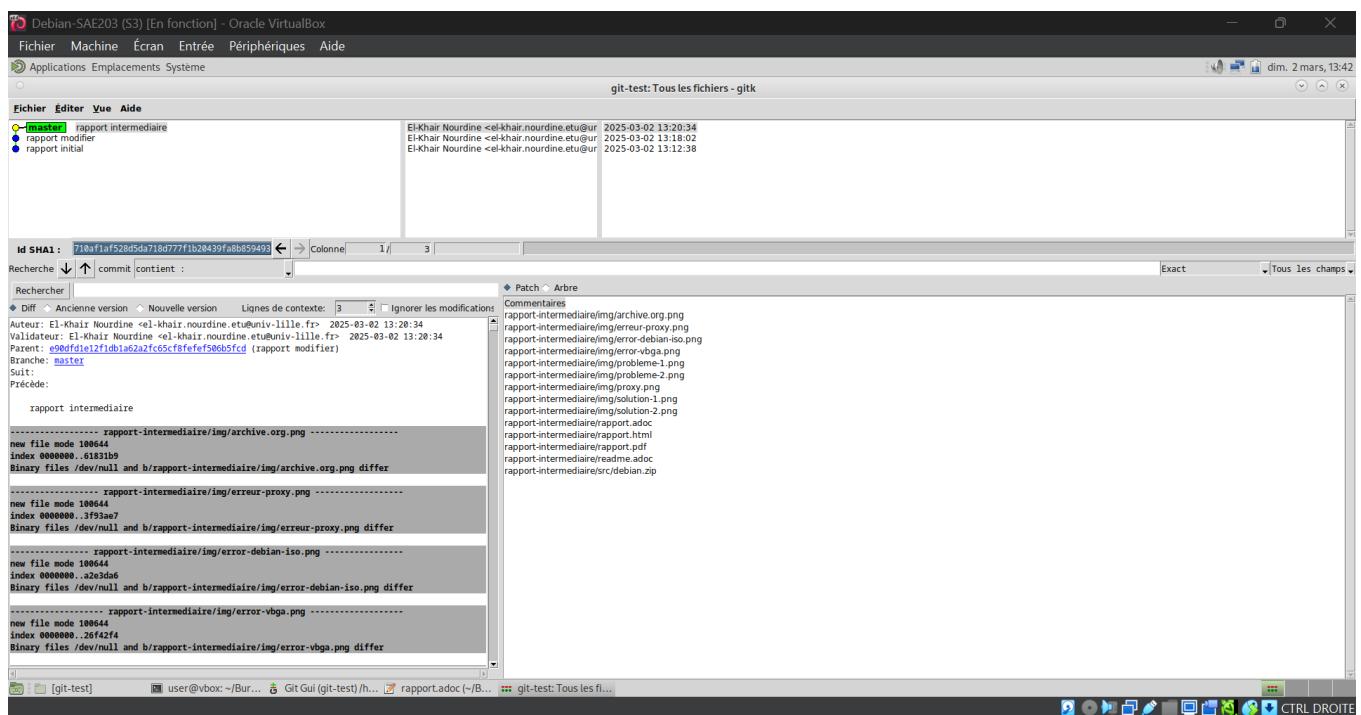
J'ai ensuite remis *rapport.adoc* à son état initial, puis cliqué sur **Recharger modifs** avant de cliquer sur **Indexer modifs** pour ajouter toutes les fichiers à l'index. Enfin, j'ai validé en commitant :



Maintenant, je vais voir ce que donne **Gitk**.

Les Interfaces Graphiques pour Git: Gitk

J'ai exécuté la commande **gitk** & pour ouvrir Gitk en arrière-plan dans le répertoire de mon rapport.



Ensuite, j'ai pu visualiser l'ensemble des commits présents dans mon dépôt en lisant les messages des commits, l'auteur, ainsi que la date de création de chaque commit.

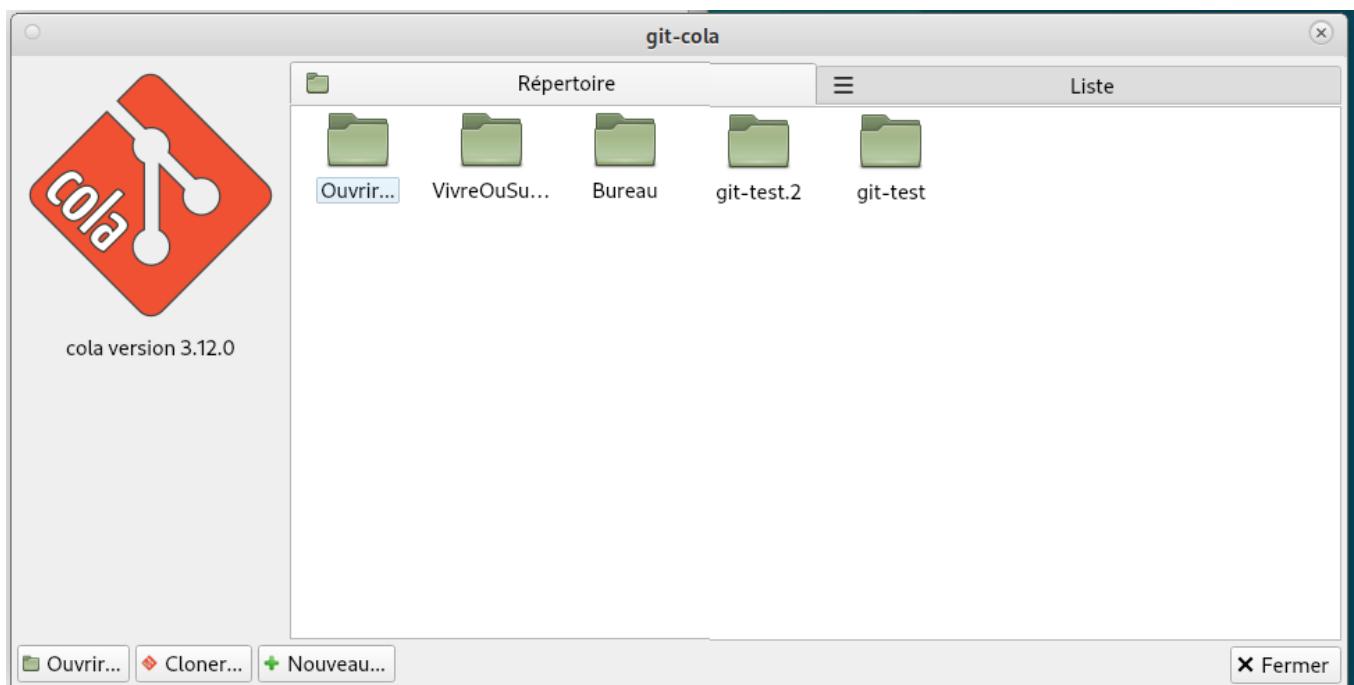
J'ai ensuite exploré l'historique en comparant les différences entre les commits, en observant l'ancienne version et la nouvelle version des fichiers.

Installons Autre Chose: Git Cola

J'ai commencé par installer **Git Cola** via la commande suivante :

```
sudo apt install git-cola
```

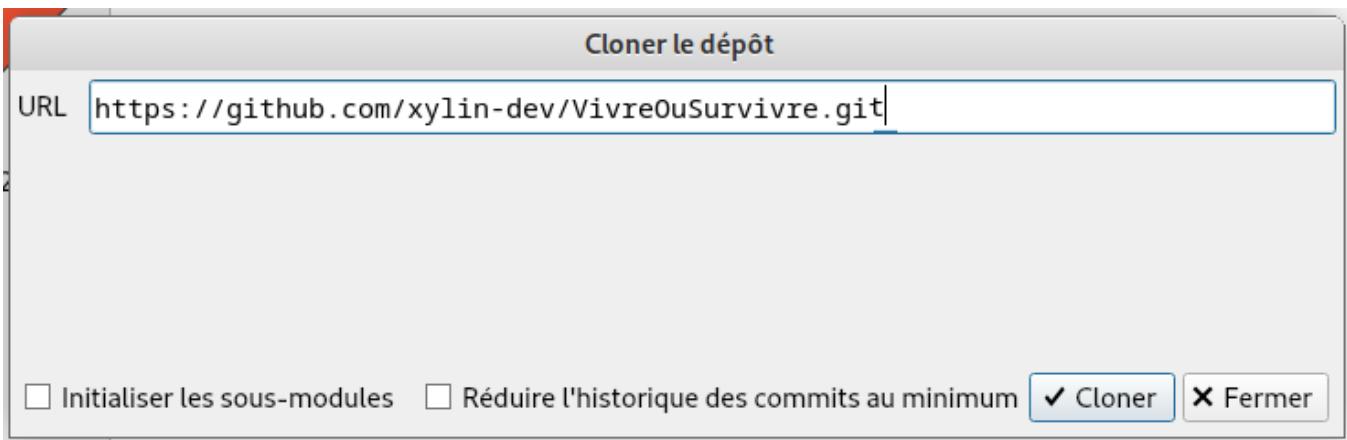
Une fois l'installation terminée, j'ai exécuté la commande suivante pour ouvrir l'interface graphique de *Git Cola* : **git cola**.



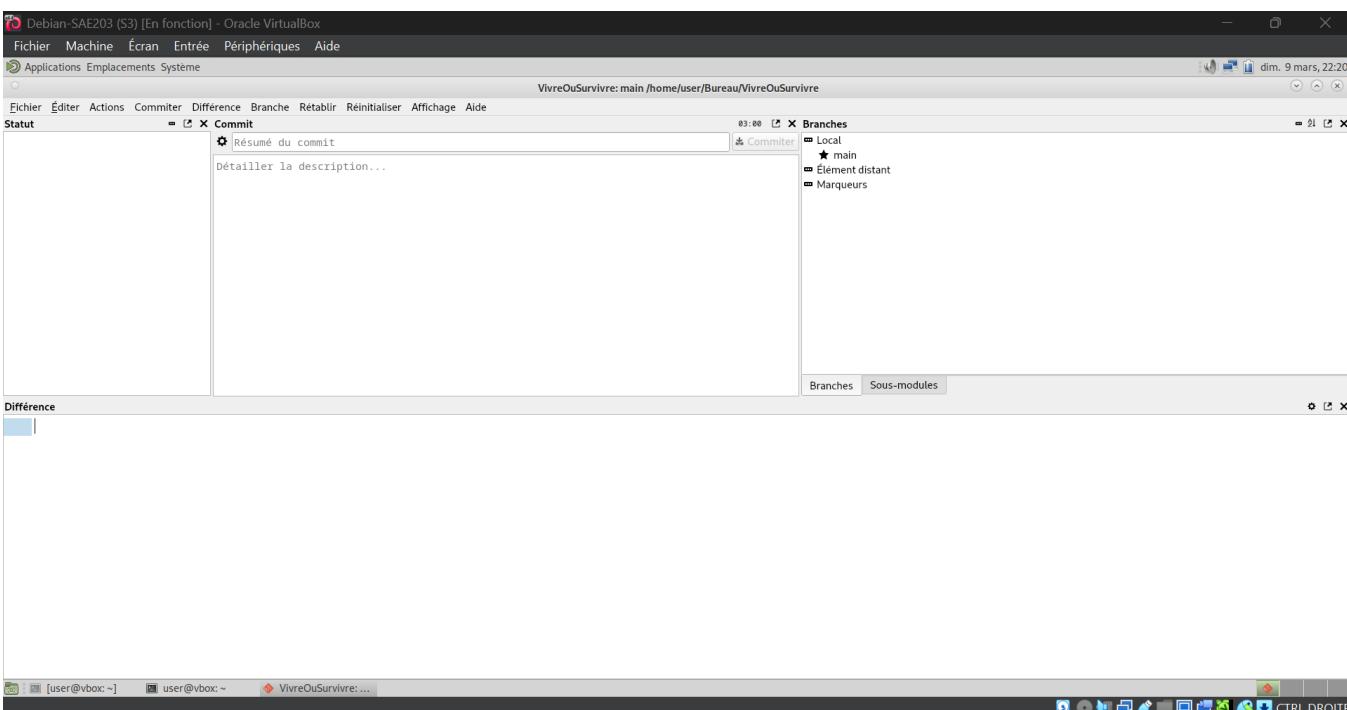
Contrairement à **Git Gui**, j'ai décidé de cloner un dépôt existant plutôt que d'utiliser un fichier présent sur mon bureau.

J'ai choisi le dépôt de la SAE 1.02 sur le logiciel ludo-pédagogique *VivreOuSurvivre*.

Pour cela, j'ai copié le lien **HTTPS** du dépôt sur [GitHub](#), je l'ai collé dans le champ correspondant, puis j'ai spécifié l'emplacement où le dépôt serait enregistré sur ma machine, à savoir sur mon bureau: [~/Bureau](#).



Après cela, l'interface de **Git Cola** s'est affichée.



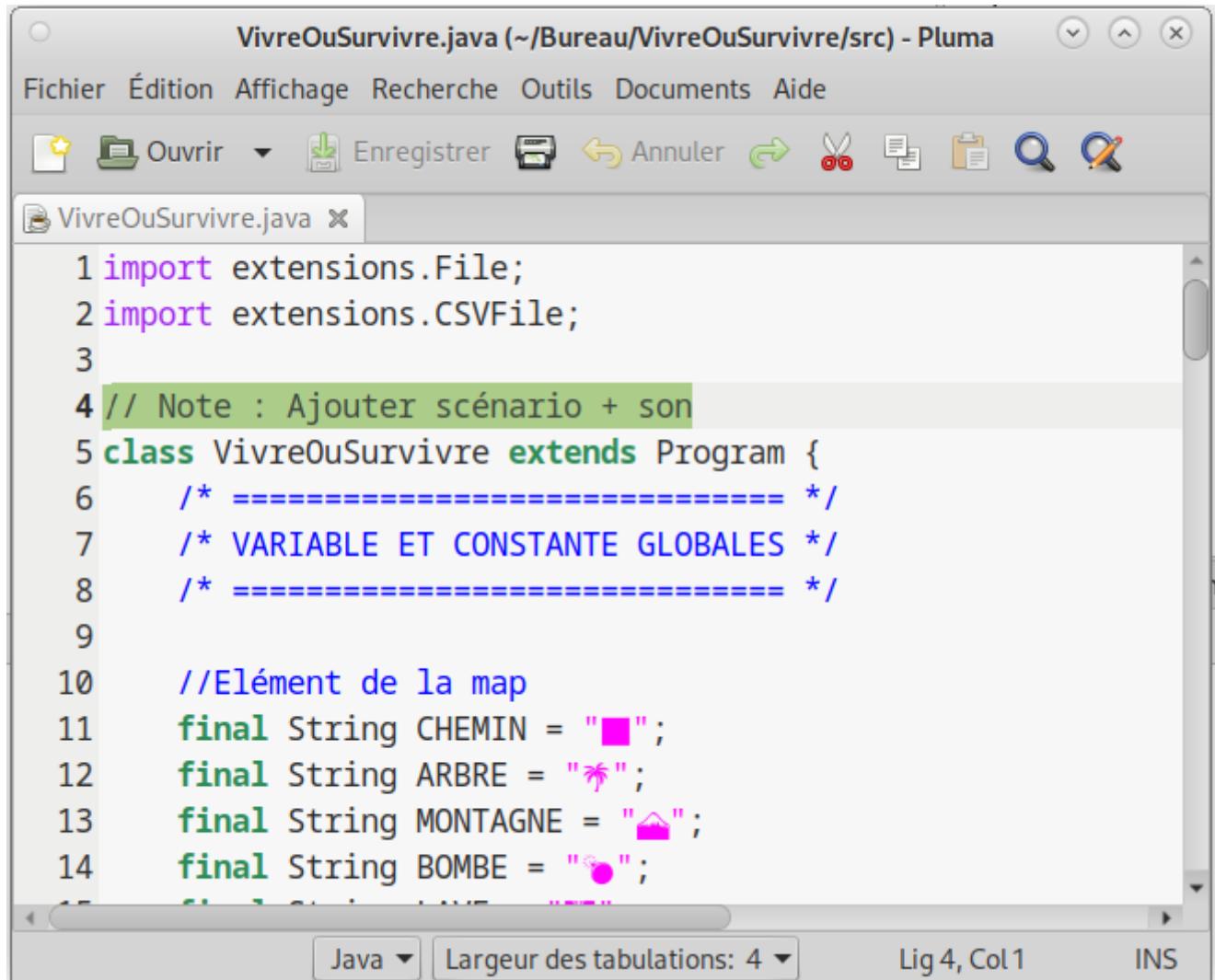
J'ai commencé à explorer l'application et ses différentes options afin de comprendre son fonctionnement et d'identifier les fonctionnalités les plus pertinentes à mentionner dans ce rapport.

Une fois familiarisé avec l'interface et après m'être documenté sur le logiciel sur leur site web [Git Cola](#), j'ai commencé à manipuler plus sérieusement les fichiers de mon dépôt.

J'ai notamment modifié le fichier source principal du dépôt à l'aide de Pluma, en exécutant la commande suivante dans le répertoire `src` où il se trouvait : `pluma VivreOuSurvivre.java`.

J'ai simplement supprimé un commentaire que j'avais oublié d'enlever avant de rendre

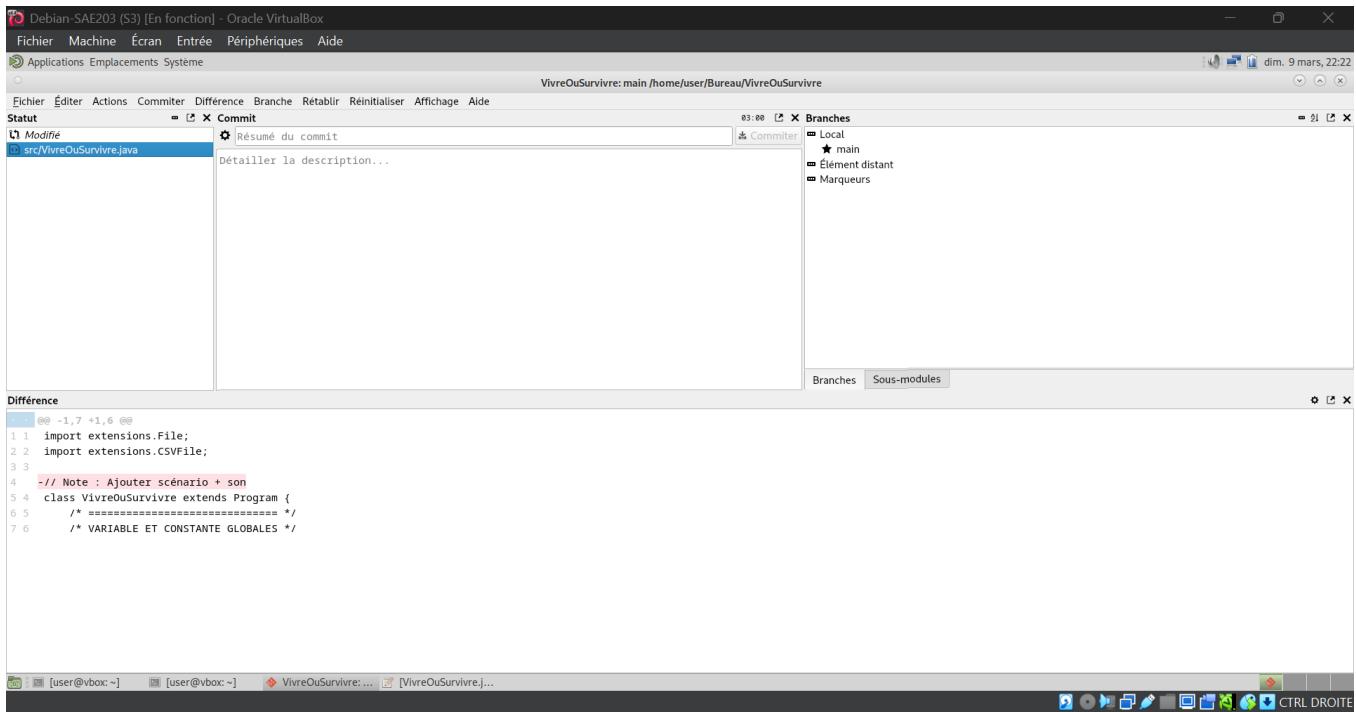
la SAE au premier semestre.



The screenshot shows the Pluma text editor interface. The title bar reads "VivreOuSurvivre.java (~/Bureau/VivreOuSurvivre/src) - Pluma". The menu bar includes Fichier, Édition, Affichage, Recherche, Outils, Documents, and Aide. Below the menu is a toolbar with icons for Ouvrir (Open), Enregistrer (Save), Annuler (Cancel), and others. The main window displays the Java code for "VivreOuSurvivre.java". The code includes imports for extensions.File and extensions.CSVFile, a class VivreOuSurvivre extending Program, and various constants defined as final String variables. The code is color-coded for syntax. A note in line 4 indicates to add a scenario and sound. The status bar at the bottom shows "Java", "Largeur des tabulations: 4", "Lig 4, Col 1", and "INS".

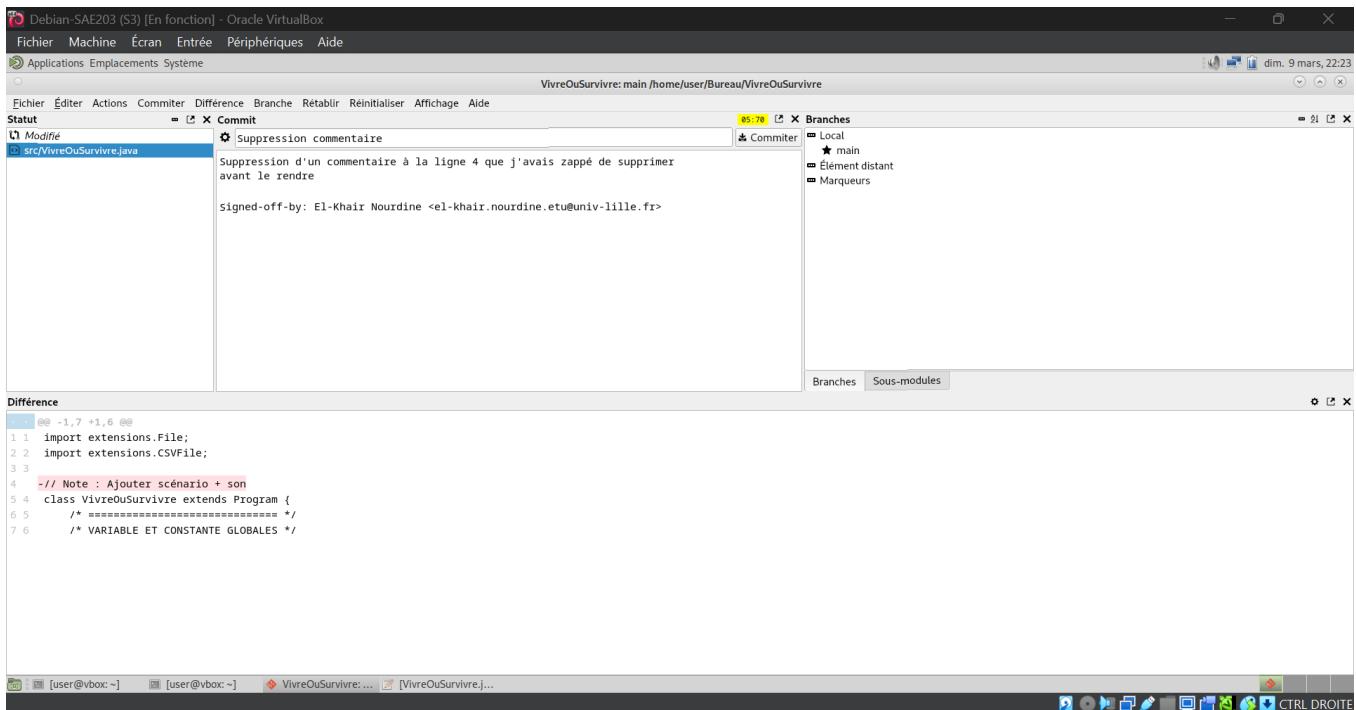
```
1 import extensions.File;
2 import extensions.CSVFile;
3
4 // Note : Ajouter scénario + son
5 class VivreOuSurvivre extends Program {
6     /* ===== */
7     /* VARIABLE ET CONSTANTE GLOBALES */
8     /* ===== */
9
10    //Elément de la map
11    final String CHEMIN = "█";
12    final String ARBRE = "♣";
13    final String MONTAGNE = "▲";
14    final String BOMBE = "●";
15    final String LAVE = "■"
16}
```

Après la modification, le fichier est apparu dans l'interface de Git Cola sans que j'aie besoin de rafraîchir l'affichage.



J'ai ensuite double-cliqué dessus afin de le **pré-committer** (*terme utilisé par Git Cola, qui correspond en réalité à l'indexation du fichier, équivalente à `git add`*).

Ensuite, j'ai procédé au commit en renseignant un message ainsi qu'un commentaire descriptif avant de valider.



J'ai ensuite voulu visualiser les différences apportées au fichier en me rendant dans le menu Définition > Branches..., qui est l'équivalent de la commande : `git diff`.

J'y ai observé la seule modification mineure que j'avais effectuée.

```

/tmp/git-blob-HcTfhU/VivreOuSurvivre.java -> /tmp/git-blob-W7qbLP/VivreOuSurvivre.java
File Edit View Global Region Line Options Display Windows Help
/tmpprojet/VivreOuSurvivre.java 1 /tmp/git-blob-W7qbLP/VivreOuSurvivre.java 1 1
import extensions.File;^M
import extensions.CSVFile;^M
^M
class VivreOuSurvivre extends Program {^M
    /* ===== */^M
    /* VARIABLE ET CONSTANTE GLOBALES */^M
    /* ===== */^M
^M
    //Elément de la map^M
    final String CHEMIN = "■";^M
    final String ARBRE = "▷";^M
    final String MONTAGNE = "▲";^M
    final String BOMBE = "●";^M
    final String LAVE = "☒";^M
    final String CARTE = "■";^M
^M
    //Chemin vers data.csv^M
    final String csvData = "ressources/CSVFile/data.csv";^M
^M
    //Couleurs du texte selon de leurs fonctions^M
    final String VERT = "\u001B[32m"; //Réussite, objectif atteint.^M
    final String ROUGE = "\u001B[31m"; //Échec, erreur.^M
    final String BLEU = "\u001B[34m"; //Important, à ne pas négliger.^M
    final String JAUNE = "\u001B[33m"; //Alerte, attention nécessaire.^M
    String RESET = "\u001B[0m"; //Couleurs et Styles par défauts^M
^M
    //Style du texte selon leurs fonctions^M
    final String GRAS = "\033[1m"; ^M
^M
    //nombre de Vie du Joueur ainsi que son nombre de Vie Precedent pour Kaomij
    int nbVie = 10;^M

```

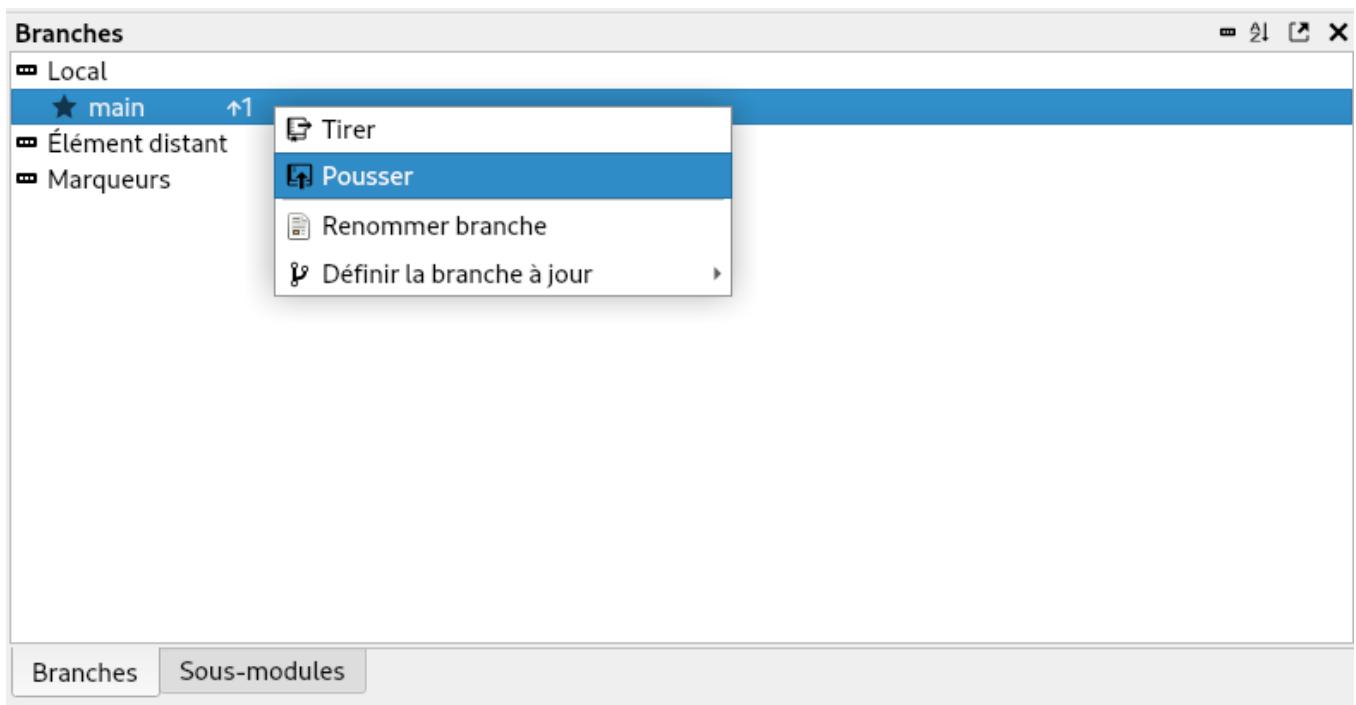
J'ai poursuivi en affichant les différentes branches du dépôt en allant dans [Branche > Visualiser la branche courante....](#)

Cette action m'a redirigé vers Gitk.

| Date | Auteur | Message |
|---------------------|---|------------------------------|
| 2025-03-09 22:24:25 | El-Khair Nourdine <el-khair.nourdine.etu@univ-lille.fr> | Initial commit |
| 2025-03-09 16:24:44 | Xylin Dev <xylindev@gmail.com> | Suppression d'un commentaire |
| 2025-01-14 22:31:02 | Xylin Dev <xylindev@gmail.com> | Suppression de la note |
| 2025-01-14 21:37:05 | Xylin Dev <xylindev@gmail.com> | Update vivreousurvivre.java |
| 2025-01-14 21:35:59 | Xylin Dev <xylindev@gmail.com> | Update vivreousurvivre.java |
| 2025-01-13 15:39:39 | Anthony <210676428+Sieonix@users.noreply.github.com> | Update vivreousurvivre.java |
| 2025-01-13 17:39:38 | Anthony <210676428+Sieonix@users.noreply.github.com> | Update vivreousurvivre.java |
| 2025-01-13 16:13:36 | Sieonix <cardosomoreira anthony@gmail.cc> | Update vivreousurvivre.java |
| 2025-01-13 15:39:49 | Sieonix <cardosomoreira anthony@gmail.cc> | Update vivreousurvivre.java |
| 2025-01-13 15:07:19 | Sieonix <cardosomoreira anthony@gmail.cc> | Update vivreousurvivre.java |

Ensuite, j'ai voulu pousser mon commit vers le **dépôt distant** sur GitHub, ce qui est l'équivalent de la commande : **git push**.

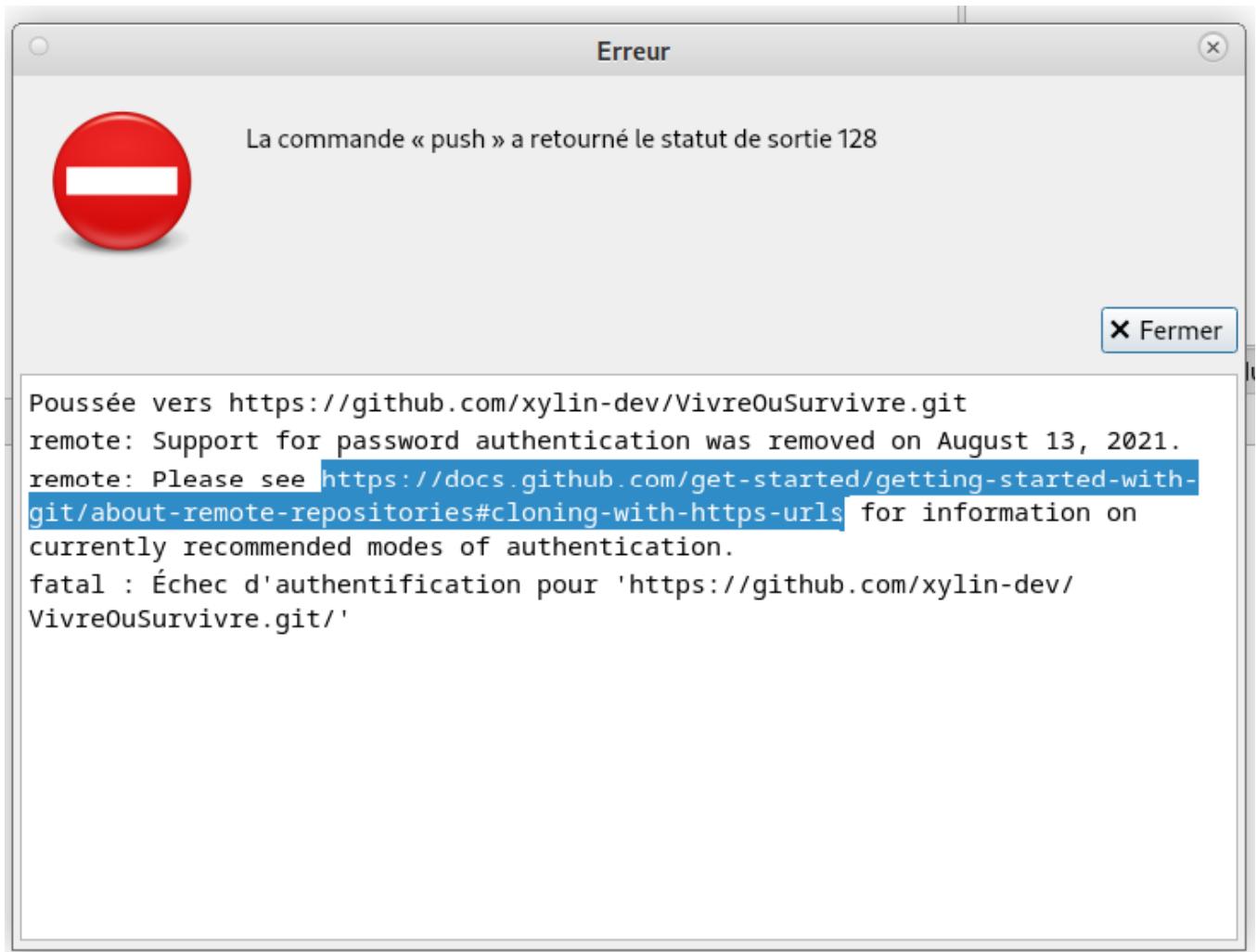
Pour cela, j'ai effectué un clic droit dans la branche *main* du dépôt local puis sélectionné **Pousser**.



Une demande d'authentification est alors apparue.



J'ai entré mon nom d'utilisateur GitHub, puis mon mot de passe, mais j'ai rencontré une erreur de push. Heureusement, un message m'indiquait où trouver de la documentation pour résoudre ce problème.



Après consultation de la documentation, j'ai compris que, depuis une mise à jour de GitHub, l'utilisation d'un mot de passe n'est plus autorisée pour l'authentification via Git en ligne de commande. À la place, il fallait générer un **Personal Access Token (PAT)**.

A screenshot of a web browser displaying the GitHub documentation at docs.github.com/fr/get-started/git-basics/about-remote-repositories#cloning-with-https-urls.

The page title is "Clonage avec des URL HTTPS".
The main content discusses using HTTPS URLs for cloning repositories, mentioning that GitHub no longer supports password authentication and instead requires a personal access token.
A "Tip" section at the bottom left suggests using a GitHub Personal Access Token if you're using SAML SSO.
A sidebar on the right lists related articles: "À propos des référentiels distants", "Création de référentiels distants", "Choix d'une URL pour votre référentiel distant", "Clonage avec des URL HTTPS", "Clonage avec des URL SSH", and "Clonage avec GitHub CLI".
The browser's address bar shows the URL, and the top navigation bar includes the GitHub logo and search bar.

J'ai donc suivi ces étapes sur GitHub :

1. Cliquer sur mon profil > Settings.
2. Faire défiler le menu latéral gauche jusqu'à Developer settings.
3. Aller dans Personal Access Tokens > Tokens (classic).
4. Cliquer sur Generate new token.
5. Générer un token nommé **sae2.03** avec les permissions adaptées: **repo**.

The screenshot shows the GitHub 'New personal access token (classic)' generation interface. On the left, there's a sidebar with options like GitHub Apps, OAuth Apps, Personal access tokens (with sub-options: Fine-grained tokens and Tokens (classic)), and a 'Preview' button. The main area has a title 'New personal access token (classic)'. It includes a note about what personal access tokens are used for, a 'Note' input field containing 'sae2.03', a 'What's this token for?' section, an 'Expiration' dropdown set to '30 days' (which will expire on Tue, Apr 8 2025), and a 'Select scopes' section. The 'repo' scope is checked, and other available scopes listed include repo:status, repo_deployment, public_repo, repo:invite, security_events, workflow, and workflow. The 'repo' scope is described as giving full control of private repositories.

Une fois le token généré et copié, j'ai réessayé d'effectuer mon push, en utilisant cette fois-ci le token au lieu du mot de passe. Cette fois, l'opération a fonctionné.

J'ai ensuite vérifié sur GitHub que mon commit était bien en ligne.

| Name | Last commit message | Last commit date |
|----------------------|-------------------------|------------------|
| .. | | |
| Joueur.java | Add files via upload | 2 months ago |
| Objectif.java | Add files via upload | 2 months ago |
| Selection.java | Add files via upload | 2 months ago |
| VivreOuSurvivre.java | Suppression commentaire | 16 minutes ago |

Enfin, pour clôturer mon exploration de Git Cola, j'ai affiché le **DAG** (*Directed Acyclic Graph*)

Graph), qui représente graphiquement l'historique des commits, en allant dans Affichage > DAG... .

The screenshot shows the Gitk graphical interface on a Linux desktop. The title bar reads "VivreOuSurvivre : main --- DAG". The interface has several panes:

- Journal**: A list of commits in the 'main' branch. The first commit is by "El-Khair N...". Subsequent commits are by "Xylin Dev.", "Anthony", and "SleelNIX".
- Differences**: A diff viewer showing changes made to the file "src/VivreOuSurvivre.java". It highlights a deletion of a comment from line 4.
- Graphe**: A vertical timeline of commits represented as nodes connected by red lines. Nodes are labeled with commit hash IDs.
- Fichiers**: A table showing file statistics: 0 additions and 1 suppression for "src/VivreOuSurvivre.java".
- Bottom Status Bar**: Shows the terminal command "[user@vbox:~]".

Semaine 4: Gitea : Installation & Utilisation du Service

4.1: Réponses aux questions.

Partie 2.0: Installation de Gitea

- **Qu'est-ce que Gitea ?**

Gitea est une solution logicielle autohébergée et complète pour le développement logiciel. Elle offre l'hébergement Git, la gestion des revues de code, la collaboration en équipe, un registre de packages ainsi que des fonctionnalités d'intégration et de déploiement continus. Légère et facile à utiliser, Gitea constitue une alternative à GitHub, Bitbucket et GitLab.^[21]

- **À quels logiciels bien connus dans ce domaine peut-on le comparer ?**

Comme mentionné précédemment, Gitea est une alternative à GitHub, Bitbucket et GitLab, **GitHub** et **Gitlab** étant les plus connus.

- **Qu'est-ce qu'un fork ?**

Un **fork** c'est quand on prends un projet existant, on le copie pour travailler dessus sans affecter l'original. C'est utile pour tester des idées ou ajouter des fonctionnalités sans risquer de casser ce qui a déjà été fait^[22].

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

Gitea est le fork du logiciel Gogs^[23] et Gogs existe encore^[24].

Partie 2.1.2: Mise à jour du binaire du service Gitea

- **Quelle version du binaire avez-vous installé ? Donnez la version et la commande permettant d'obtenir cette information.**

En étant dans le répertoire contenant le fichier binaire de Gitea, j'exécute la commande : `./gitea -v`. La commande exécutée retourne le message suivant : "Gitea version 1.23.5 built with GNU Make 4.3, go1.23.7 : bindata, sqlite, sqlite_unlock_notify".

Conclusion, la version du binaire est la 1.23.5 ^[25].

- **Comment faire pour mettre à jour le binaire de votre service sans devoir tout reconfigurer ? Essayez en mettant à jour vers la version 1.22-dev.**

Au moment de rédiger ce rapport, je n'ai pas trouvé la version **1.22-dev** en effectuant une recherche avec Ctrl+F sur [le site de téléchargement de Gitea](#).

En cherchant des versions "dev", je n'ai trouvé que les versions **1.7.0-dev**, **1.6.0-dev** et

1.5.0-dev. Par conséquent, je vais me permettre de faire l'installation et la mise à jour avec la version **1.22.0**.

- Pour ce faire, je me réfère à la section [Upgrade from binary](#) de la documentation officielle de Gitea.
1. Je télécharge l'ancienne version de Gitea (1.22.0) en exécutant la commande suivante : `wget -O gitea-1.22.0 https://dl.gitea.com/gitea/1.22.0/gitea-1.22.0-linux-amd64`. Cette commande est lancée dans le répertoire `/var/lib/gitea`, où se trouve actuellement le fichier binaire de la version 1.23.5 et en étant sur l'utilisateur `git`.
 2. J'arrête le processus actuel de Gitea (*car j'ai utilisé la commande `./gitea web & pour lancer Gitea en arrière-plan`*) en utilisant la commande suivante pour récupérer son PID : `ps aux | grep gitea` (*cette commande permet de lister les processus en cours et de trouver celui lié à Gitea*). Ensuite, pour arrêter ce processus, j'utilise la commande `kill -9` suivie du PID correspondant (dans mon cas, le PID est 4201) : `kill -9 4201` (*cela termine le processus de Gitea avant de procéder à l'installation de la nouvelle version*). Pour plus de détails sur la commande `ps aux`, je me suis référé au site de [Malekal](#).
 3. Enfin, j'exécute la commande suivante pour lancer Gitea avec la version 1.22.0 : `./gitea-1.22.0 web`. Cependant, cette tentative échoue, car il y a un problème de compatibilité avec la base de données. Le message d'erreur suivant apparaît : **Migration Error: Your database (migration version: 312) is for a newer Gitea, you can not use the newer database for this old Gitea release (299)**.
 4. Du coup, je décide de retenter l'opération avec la **version 1.24.3 amd64** de Gitea en utilisant les mêmes procédés, et cette fois-ci, cela fonctionne.

Partie 2.3: Pour aller plus loin

- Qu'est-ce que l'intégration et la livraison continue ?
 - **CI (*intégration continue*)**: C'est quand on fusionne régulièrement notre code avec celui des autres dans la branche principale, et ça déclenche des tests automatiques pour vérifier que tout fonctionne bien ensemble^[26].
 - **CD (*livraison continue*)** : C'est un peu la suite de l'intégration continue. Une fois que notre code a passé les tests, il peut être automatiquement déployé sur le serveur ou en production^[27].

4.2: Rapport sur nos actions réalisées durant cette semaine.

Le rapport sera structuré en deux parties :

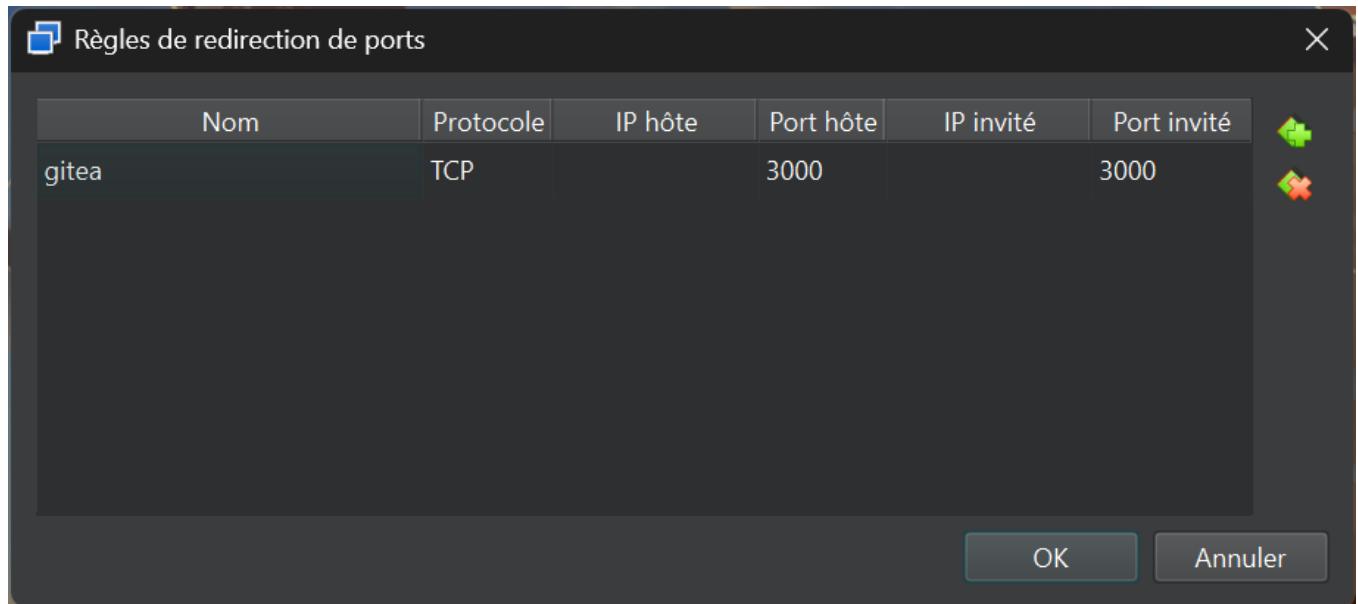


- **L'installation de Gitea** où je décris le processus d'installation de Gitea, les erreurs rencontrées et les solutions que j'ai appliquées pour les corriger.
- **L'utilisation de Gitea** où j'explique mon utilisation de Gitea.

Installation de Gitea

J'ai commencé par configurer la redirection de port en allant dans la "**Configuration**" de ma machine virtuelle, section "**Réseau**".

J'ai cliqué sur "**Redirection de port**" et ajouté le port **3000** aussi bien sur la machine hôte que sur l'invité.



Je n'ai pas besoin de toucher au **mode d'accès réseau**, qui était par défaut en NAT. Ensuite, j'ai démarré ma machine pour commencer l'installation de Gitea.

Pendant que la machine démarrait, j'en ai profité pour lire la [documentation](#) sur l'installation de Gitea via le fichier binaire.

J'ai préféré la méthode de téléchargement avec `wget` plutôt que via le navigateur, car c'est plus rapide.

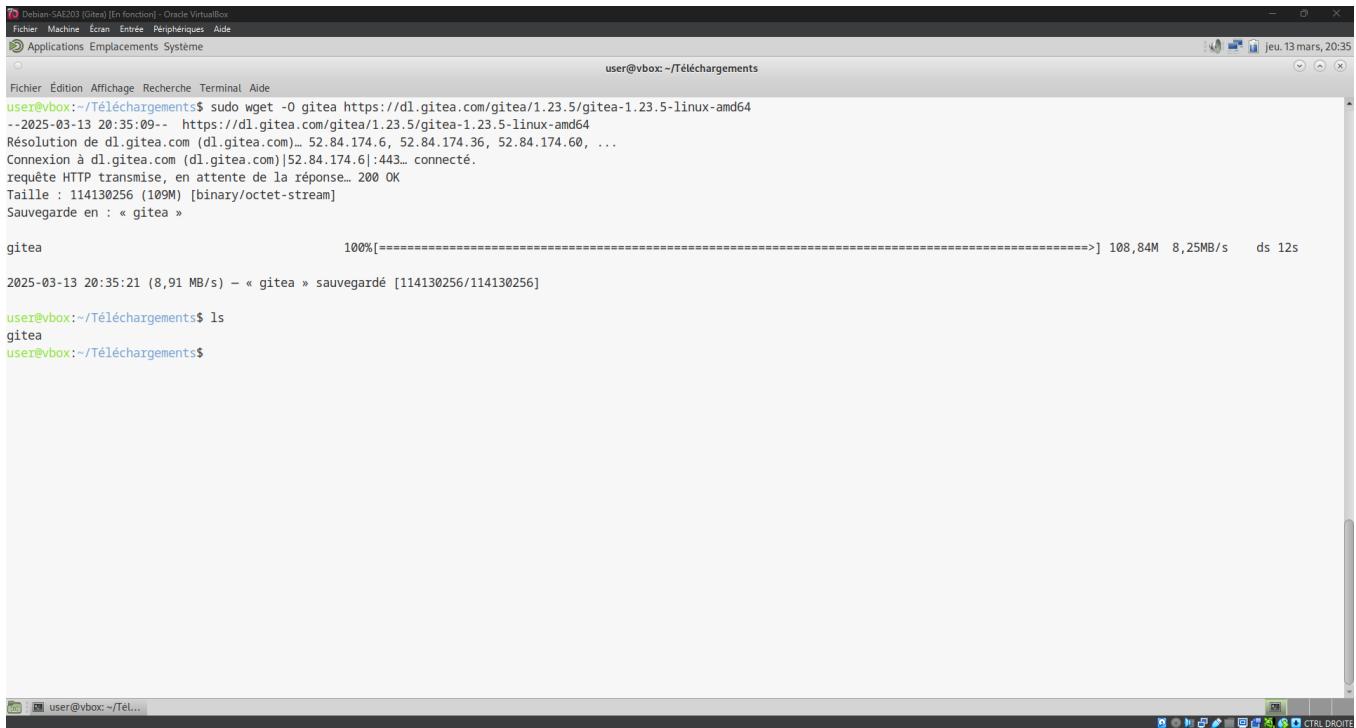
Une fois la machine démarrée, j'ai ouvert le Terminal et me suis déplacé vers le dossier Téléchargements avec `cd Téléchargements/`.

Ensuite, je suis allé sur la page de [téléchargement](#) de Gitea, où j'ai pris la dernière version disponible (*1.23.5*).

J'ai cherché la version compatible avec mon système (*1.23.5-linux-amd64 pour du 64-bit*), fait un clic droit dessus et copié son lien.

Dans mon terminal, j'ai téléchargé le fichier avec :

```
# Commande qui télécharge le fichier gitea-1.23.5-linux-amd64 depuis l'URL donnée et le sauvegarde sous le nom gitea.  
wget -O gitea https://dl.gitea.com/gitea/1.23.5/gitea-1.23.5-linux-amd64
```



```
Debian-SAC203 [Gitea] [En fonction] - Oracle VirtualBox
Fichier Machine Écran Entrée Périphériques Aide
Applications Emplacements Système
Fichier Édition Affichage Recherche Terminal Aide
user@vbox:~/Téléchargements$ sudo wget -O gitea https://dl.gitea.com/gitea/1.23.5/gitea-1.23.5-linux-amd64
--2025-03-13 20:35:09-- https://dl.gitea.com/gitea/1.23.5/gitea-1.23.5-linux-amd64
Résolution de dl.gitea.com (dl.gitea.com)... 52.84.174.6, 52.84.174.36, 52.84.174.60, ...
Connexion à dl.gitea.com (dl.gitea.com)|52.84.174.6|:443... connecté.
requête HTTP transmise, en attente de la réponse... 200 OK
Taille : 114130256 (109M) [binary/octet-stream]
Sauvegarde en : « gitea »

gitea                                100%[=====] 108,84M  8,25MB/s   ds 12s

2025-03-13 20:35:21 (8,91 MB/s) - « gitea » sauvegardé [114130256/114130256]

user@vbox:~/Téléchargements$ ls
gitea
user@vbox:~/Téléchargements$
```

Téléchargement du fichier binaire de Gitea

Après l'installation, j'ai rendu le fichier exécutable en utilisant la commande **chmod +x gitea**.



wget est un outil GNU en ligne de commande qui permet de télécharger des fichiers via **HTTP**, **HTTPS** et **FTP**. L'option **-O** me permet d'enregistrer le fichier sous un nom spécifique plutôt que celui par défaut (*gitea-1.23.5-linux-amd64*).

Ensuite, j'ai suivi la documentation sur la **signature GPG** pour vérifier l'authenticité du fichier.



Gitea signe ses binaires avec une clé GPG pour garantir qu'ils n'ont pas été modifiés.

J'ai récupéré la clé publique avec :

```
# Commande qui demande à GPG de récupérer la clé publique spécifique (identifiée par l'ID 7C9E68152594688862D62AF62D9AE806EC1592E2) depuis le serveur de clefs keys.openpgp.org.
```

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

```
user@vbox: ~/Téléchargements
Fichier Édition Affichage Recherche Terminal Aide
user@vbox:~/Téléchargements$ gpg --keyserver keys.openpgp.org --recv 7C9E68152594688862D62AF62D9AE806EC1592E2
gpg: répertoire « /home/user/.gnupg » créé
gpg: le trousseau local « /home/user/.gnupg/pubring.kbx » a été créé
gpg: /home/user/.gnupg/trustdb.gpg : base de confiance créée
gpg: clef 2D9AE806EC1592E2 : clef publique « Teabot <teabot@gitea.io> » importée
gpg:      Quantité totale traitée : 1
gpg:                      importées : 1
user@vbox:~/Téléchargements$
```

Puis, j'ai téléchargé le fichier **.asc** associé au binaire, de la même manière que [gitea](#):

```
# Commande qui télécharge le fichier gitea-1.23.5-linux-amd64.asc depuis l'URL donnée et le sauvegarde sous le nom gitea-asc.
```

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

```
Débian-SAGE203 [Gitea] [En Fonction] - Oracle VirtualBox
Fichier Machine Écran Entrée Périphériques Aide
Applications Emplacements Système
Fichier Édition Affichage Recherche Terminal Aide
user@vbox:~/Téléchargements$ wget -O gitea-asc https://dl.gitea.com/gitea/1.23.5/gitea-1.23.5-linux-amd64.asc
--2025-03-13 20:40:36-- https://dl.gitea.com/gitea/1.23.5/gitea-1.23.5-linux-amd64.asc
Résolution de dl.gitea.com (dl.gitea.com)... 52.84.174.33, 52.84.174.36, 52.84.174.60, ...
Connexion à dl.gitea.com (dl.gitea.com)|52.84.174.33|:443... connecté.
requête HTTP transmise, en attente de la réponse... 200 OK
Taille : 566 [application/pgp-keys]
Sauvegarde en : « gitea-asc »

gitea-asc          100%[=====] 566  --.-KB/s   ds 0s
2025-03-13 20:40:36 (7,19 MB/s) - « gitea-asc » sauvegardé [566/566]

user@vbox:~/Téléchargements$ ls
gitea  gitea-asc
user@vbox:~/Téléchargements$
```



Un fichier ASC est une variante du format ASCII, qui est un fichier de cryptage utilisé par Pretty Good Privacy (PGP) pour la protection des communications en ligne.^[28]

J'ai vérifié la signature avec la commande **gpg --verify gitea-asc gitea**.

Selon la documentation si le message renvoyé indique "Bonne signature de « Teabot teabot@gitea.io »", alors tout est bon, malgré l'éventuel avertissement affiché.

```

user@vbox:~/Téléchargements$ gpg --verify gitea-asc gitea
gpg: Signature faite le mer. 05 mars 2025 01:09:17 CET
gpg:           avec la clef RSA CC64B1DB67ABEBCAB24B6455FC346329753F4B0
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
user@vbox:~/Téléchargements$
```

Après ça, j'ai vérifié que ma version de git était bien supérieure à 2.0 avec `git -v`, j'ai 2.39.5.

Ensuite, j'ai tenté d'ajouter l'utilisateur git, mais la commande a échoué car elle nécessitait sudo ou d'être en root.

```

user@vbox:~/Téléchargements$ sudo adduser --system --shell /bin/bash --gecos 'Git Version Control' --group --disabled-password --home /home/git git
Ajout de l'utilisateur système « git » (UID 112) ...
Ajout du nouveau groupe « git » (GID 122) ...
Ajout du nouvel utilisateur « git » (UID 112) avec pour groupe d'appartenance « git » ...
Création du répertoire personnel « /home/git » ...
user@vbox:~/Téléchargements$
```

```
# Commande qui crée un utilisateur système git avec un répertoire home /home/git, un shell Bash, et un groupe associé git. Le mot de passe est désactivé, ce qui signifie que l'utilisateur ne pourra pas se connecter directement avec un mot de passe.

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

Même problème pour la création des répertoires nécessaires à Gitea :

```
# Crée les répertoires nécessaires à Gitea
mkdir -p /var/lib/gitea/{custom,data,log}

# Attribue les permissions appropriées à l'utilisateur 'git'
chown -R git:git /var/lib/gitea/

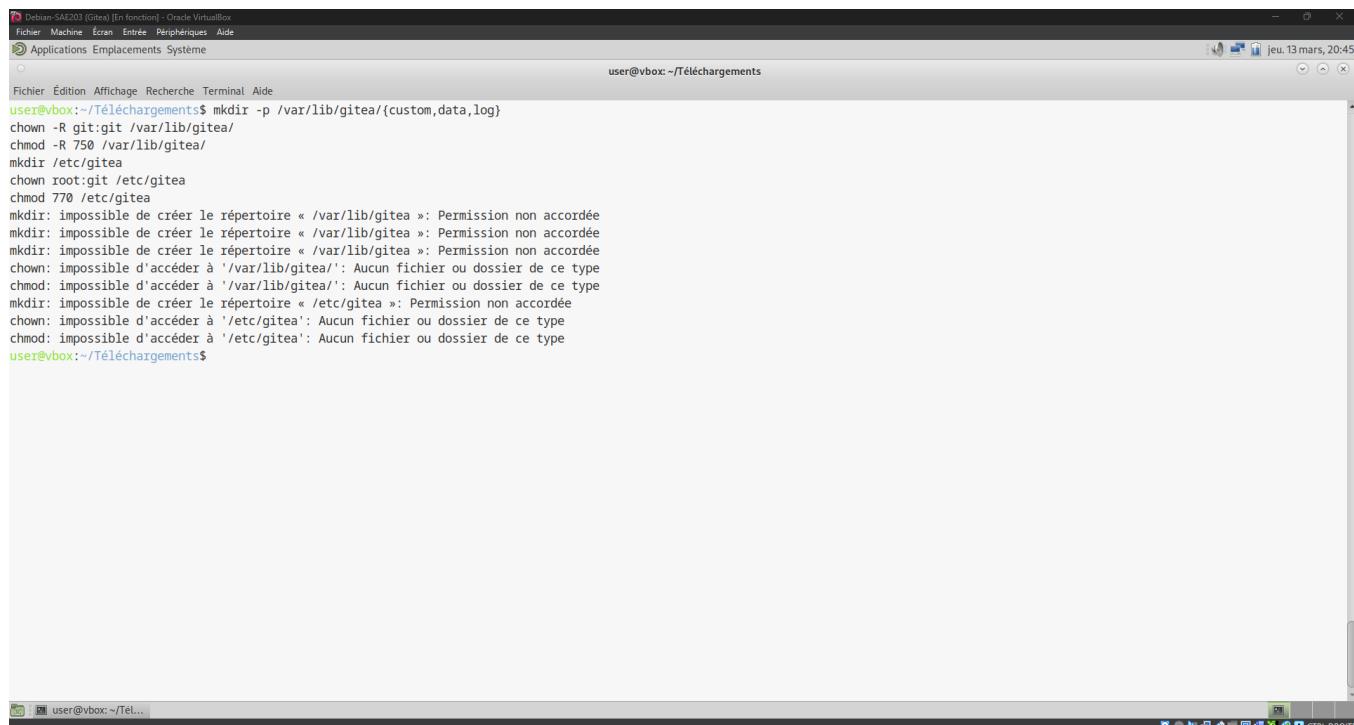
# Modifie les permissions des répertoires pour la sécurité
chmod -R 750 /var/lib/gitea/

# Crée le répertoire de configuration de Gitea
mkdir /etc/gitea

# Modifie les permissions du répertoire de configuration
chown root:git /etc/gitea/
```

```
# Assure que seul l'utilisateur 'git' et 'root' aient accès
chmod 770 /etc/gitea
```

Sans **sudo**, j'avais plein d'erreurs.



The screenshot shows a terminal window titled "Debian-SAE203 (Gitea) [En fonction] - Oracle VirtualBox". The command entered was:

```
user@vbox:~/Téléchargements$ mkdir -p /var/lib/gitea/{custom,data,log}
chown -R git:git /var/lib/gitea/
chmod -R 750 /var/lib/gitea/
mkdir /etc/gitea
chown root:git /etc/gitea
chmod 770 /etc/gitea
mkdir: impossible de créer le répertoire « /var/lib/gitea »: Permission non accordée
mkdir: impossible de créer le répertoire « /var/lib/gitea »: Permission non accordée
mkdir: impossible de créer le répertoire « /var/lib/gitea »: Permission non accordée
chown: impossible d'accéder à '/var/lib/gitea': Aucun fichier ou dossier de ce type
chmod: impossible d'accéder à '/var/lib/gitea': Aucun fichier ou dossier de ce type
mkdir: impossible de créer le répertoire « /etc/gitea »: Permission non accordée
chown: impossible d'accéder à '/etc/gitea': Aucun fichier ou dossier de ce type
chmod: impossible d'accéder à '/etc/gitea': Aucun fichier ou dossier de ce type
user@vbox:~/Téléchargements$
```

À ce moment-là, j'étais un peu bloqué, car la documentation n'était pas super explicite sur certains points.

J'ai donc exploré d'autres sections et découvert la commande **gitea web** ^[29].

J'ai voulu l'exécuter, mais ça ne marchait pas car Gitea n'était pas installé.

Il fallait d'abord exécuter le fichier binaire avec : **./gitea web**

Mais après 15 minutes, j'avais oublié certaines commandes utilisées précédemment et je ne faisais plus le lien entre ce que disait la doc et ce que j'avais fait.

J'ai donc lancé **./gitea web** depuis le répertoire **Téléchargements/**, ce qui était, spoiler, une erreur.

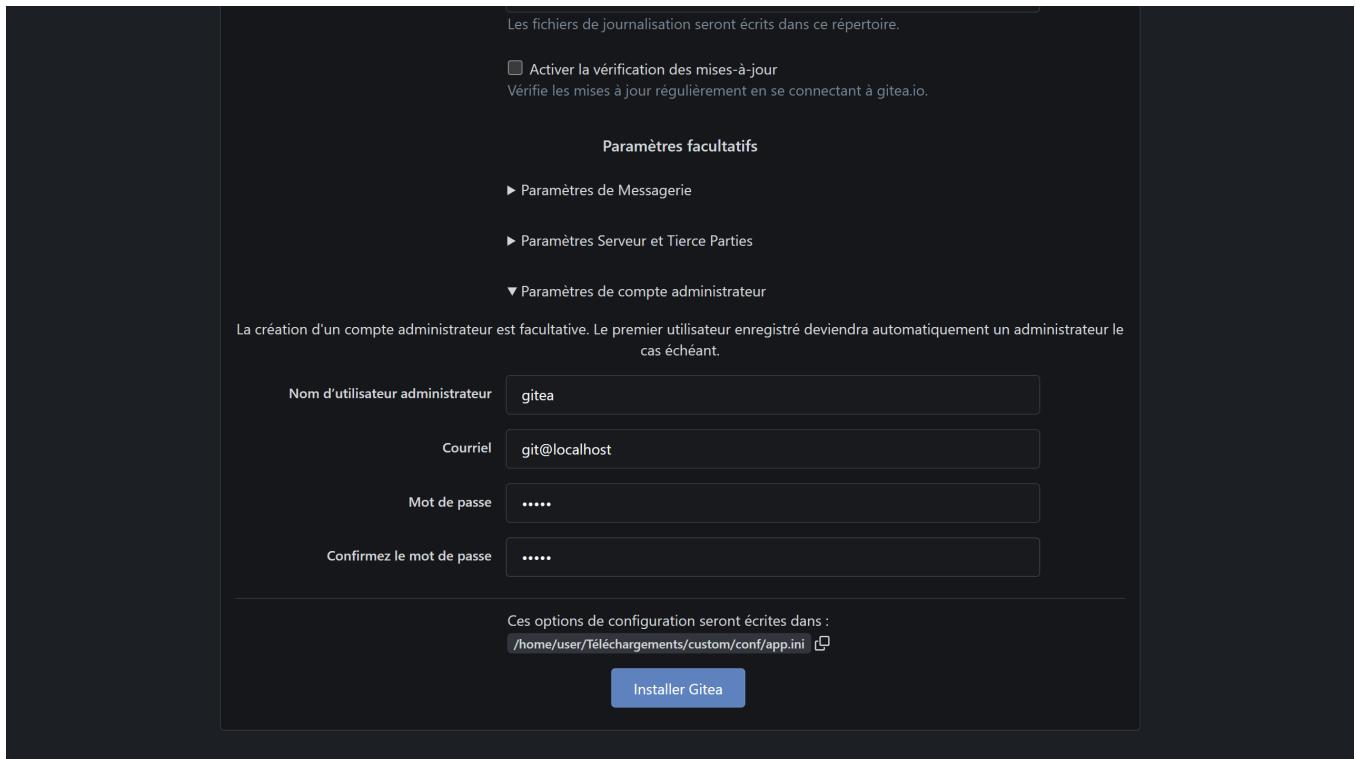
```

Debian-SAE2.03 [Gitea] [En fonction] - Oracle VirtualBox
Fichier Machine Écran Entrée Périphériques Aide
Applications Emplacements Système
Fichier Édition Affichage Recherche Terminal Aide
user@vbox:~/Téléchargements$ ./gitea web
2025/03/13 20:56:46 cmd/web.go:253:runWeb() [I] Starting Gitea on PID: 3522
2025/03/13 20:56:46 cmd/web.go:112:showWebStartupMessage() [I] Gitea version: 1.23.5 built with GNU Make 4.3, go1.23.7 : bindata, sqlite, sqlite_unlock_notify
2025/03/13 20:56:46 cmd/web.go:113:showWebStartupMessage() [I] * RunMode: prod
2025/03/13 20:56:46 cmd/web.go:114:showWebStartupMessage() [I] * AppPath: /home/user/Téléchargements/gitea
2025/03/13 20:56:46 cmd/web.go:115:showWebStartupMessage() [I] * WorkPath: /home/user/Téléchargements
2025/03/13 20:56:46 cmd/web.go:116:showWebStartupMessage() [I] * CustomPath: /home/user/Téléchargements/custom
2025/03/13 20:56:46 cmd/web.go:117:showWebStartupMessage() [I] * ConfigFile: /home/user/Téléchargements/custom/conf/app.ini
2025/03/13 20:56:46 cmd/web.go:118:showWebStartupMessage() [I] Prepare to run install page
2025/03/13 20:56:47 cmd/web.go:315:listen() [I] Listen: http://0.0.0.0:3000
2025/03/13 20:56:47 cmd/web.go:319:listen() [I] AppURL(ROOT_URL): http://localhost:3000/
2025/03/13 20:56:47 .../graceful/server.go:50>NewServer() [I] Starting new Web server: tcp:0.0.0.0:3000 on PID: 3522

```

J'ai commencé à remplir les champs selon les consignes du PDF:

| Paramètres de la base de données | |
|--|--|
| Type de base de données * | SQLite3 |
| Emplacement * | /home/user/Téléchargements/data/gitea.db |
| Configuration générale | |
| Titre du site * | Gitea - SAE2.03 |
| Emplacement racine des dépôts * | |
| Emplacement racine des dépôts * | /home/user/Téléchargements/data/gitea-repositories |
| Répertoire racine Git LFS | /home/user/Téléchargements/data/lfs |
| Exécuter avec le compte d'un autre utilisateur * | user |
| Domaine du serveur * | localhost |



Au moment de cliquer sur "Installer Gitea", j'ai remarqué un détail qui m'a fait tilt :

Ces options de configuration seront écrites dans :
/home/user/Téléchargements/custom/conf/app.ini ↗

En fait, la documentation mentionnait **app.ini**, qui devait se trouver dans **/etc/gitea**, j'ai compris que je faisais n'importe quoi, mais j'ai aussi compris comment procéder.

J'ai donc restauré un instantané de ma machine et pris du recul.

Après une relecture attentive de la documentation (*exploit perso, à 21h mon cerveau refuse généralement de fonctionner*), j'ai compris que lancer gitea depuis Téléchargements/ était une erreur.

La commande `sudo mkdir -p /var/lib/gitea/{custom,data,log}` montrait où je devais mettre le fichier binaire. J'ai donc déplacé le binaire au bon endroit : `mv gitea /var/lib/gitea`.

Cette fois, en exécutant : `sudo /var/lib/gitea/gitea web`, j'ai eu une erreur.

```
user@serveur:~$ sudo /var/lib/gitea/gitea web
2025/03/15 15:42:17 ...s/setting/setting.go:179:loadRunModeFrom() 🔴 Gitea is not supposed to be run as root. Sorry. If you need to use privileged TCP ports please instead use setcap and the 'cap_net_bind_service' permission
```

Après un moment de réflexion, je suis retourné voir la doc et j'ai **(re)découvert** que l'utilisateur **git** devait avoir les droits sur le répertoire : `chown -R git:git`

/var/lib/gitea/



`chown` est une commande qui permet de changer le propriétaire d'un fichier sur Linux.

Je devais donc exécuter gitea en tant que `git`.

J'ai cherché comment changer d'utilisateur en ligne de commande sous Linux.

Une recherche m'a mené à cet article : [Comment changer d'utilisateur dans la ligne de commande Linux](#).

J'ai tenté `su - git`, mais on m'a demandé un mot de passe (*alors que git n'en avait pas*).

```
user@serveur:~$ su - git
Mot de passe :
su: Échec de l'authentification
```

Après plusieurs essais (root, user, git...), rien ne fonctionnait. En relisant, l'article, je suis tombé sur la bonne méthode :

```
# Commande permettant à l'utilisateur courant _(qui doit avoir des droits sudo)_ de devenir
# l'utilisateur git et de charger l'environnement de cet utilisateur.
sudo su - git
```

Ça a marché !

Je suis allé dans `/var/lib/gitea/` avec `cd /var/lib/gitea/` et ai exécuté : `./gitea server`.

```

git@serveur:~$ cd /var/lib/gitea
git@serveur:/var/lib/gitea$ ./gitea web
2025/03/15 16:02:20 cmd/web.go:253:runWeb() [I] Starting Gitea on PID: 6447
2025/03/15 16:02:20 cmd/web.go:112:showWebStartupMessage() [I] Gitea version: 1.23.5 built with GNU Make 4.3, go1.23.7 : bindata, sqlite, sqlite_unlock_notify
2025/03/15 16:02:20 cmd/web.go:113:showWebStartupMessage() [I] * RunMode: prod
2025/03/15 16:02:20 cmd/web.go:114:showWebStartupMessage() [I] * AppPath: /var/lib/gitea/gitea
2025/03/15 16:02:20 cmd/web.go:115:showWebStartupMessage() [I] * WorkPath: /var/lib/gitea
2025/03/15 16:02:20 cmd/web.go:116:showWebStartupMessage() [I] * CustomPath: /var/lib/gitea/custom
2025/03/15 16:02:20 cmd/web.go:117:showWebStartupMessage() [I] * ConfigFile: /var/lib/gitea/custom/conf/app.ini
2025/03/15 16:02:20 cmd/web.go:118:showWebStartupMessage() [I] Prepare to run web server
2025/03/15 16:02:20 routers/init.go:117:InitWebInstalled() [I] Git version: 2.39.5 (home: /var/lib/gitea/data/home)
2025/03/15 16:02:20 routers/init.go:119:InitWebInstalled() [W] sha256 hash support is disabled - requires Git >= 2.42.
2025/03/15 16:02:21 ...s/storage/storage.go:176:initAttachments() [I] Initialising Attachment storage with type: local
2025/03/15 16:02:21 ...les/storage/local.go:33>NewLocalStorage() [I] Creating new Local Storage at /var/lib/gitea/data/attachments
2025/03/15 16:02:21 ...s/storage/storage.go:166:initAvatars() [I] Initialising Avatar storage with type: local
2025/03/15 16:02:21 ...les/storage/local.go:33>NewLocalStorage() [I] Creating new Local Storage at /var/lib/gitea/data/avatars
2025/03/15 16:02:21 ...s/storage/storage.go:192:initRepoAvatars() [I] Initialising Repository Avatar storage with type: local
2025/03/15 16:02:21 ...les/storage/local.go:33>NewLocalStorage() [I] Creating new Local Storage at /var/lib/gitea/data/repo-avatars
2025/03/15 16:02:21 ...s/storage/storage.go:186:initLFS() [I] Initialising LFS storage with type: local
2025/03/15 16:02:21 ...les/storage/local.go:33>NewLocalStorage() [I] Creating new Local Storage at /var/lib/gitea/data/lfs
2025/03/15 16:02:21 ...s/storage/storage.go:198:initRepoArchives() [I] Initialising Repository Archive storage with type: local
2025/03/15 16:02:21 ...les/storage/local.go:33>NewLocalStorage() [I] Creating new Local Storage at /var/lib/gitea/data/repo-archive
2025/03/15 16:02:21 ...s/storage/storage.go:208:initPackages() [I] Initialising Packages storage with type: local
2025/03/15 16:02:21 ...les/storage/local.go:33>NewLocalStorage() [I] Creating new Local Storage at /var/lib/gitea/data/packages
2025/03/15 16:02:21 ...s/storage/storage.go:219:initActions() [I] Initialising Actions storage with type: local
2025/03/15 16:02:21 ...les/storage/local.go:33>NewLocalStorage() [I] Creating new Local Storage at /var/lib/gitea/data/actions_log
2025/03/15 16:02:21 ...s/storage/storage.go:223:initActions() [I] Initialising ActionsArtifacts storage with type: local
2025/03/15 16:02:21 ...s/storage/storage.go:223:initActions() [I] Creating new Local Storage at /var/lib/gitea/data/actions_artifacts

```

La capture d'écran ci-dessus montre la commande après installation.



J'ai oublié de capturer celle pendant l'installation, mais ça devait être similaire à celles [plus haut](#).

Enfin, j'ai suivi les consignes de la semaine concernant **sqlite3** et **admin** puis j'ai installé Gitea.

- **Petite incohérence dans la documentation (je pense) :**

Selon la documentation, il faut sécuriser **/etc/gitea/** avec : **chmod 750 /etc/gitea**.

Mais le fichier **app.ini** censé s'y trouver n'existe pas.

```

git@serveur:~$ chmod 640 /etc/gitea/app.ini
chmod: impossible d'accéder à '/etc/gitea/app.ini': Aucun fichier ou dossier de ce type

```

J'ai donc sécurisé celui qui était dans **/var/lib/gitea/custom/conf/** avec :

```

# Commande permettant de protéger les fichiers de configuration sensibles (comme app.ini pour Gitea) en limitant l'accès en écriture au propriétaire, tout en permettant au groupe de lire ces fichiers pour les consulter si nécessaire.

```

```
chmod 640 /var/lib/gitea/custom/conf/app.ini
```

Et pour être sûr :

```

# Commande permettant de sécuriser l'accès aux fichiers de configuration de Gitea en restreignant l'accès aux utilisateurs non autorisés tout en permettant à l'utilisateur et au groupe spécifiés de gérer ces fichiers.

```

```
chmod 750 /var/lib/gitea/custom/conf/
```

Utilisation de Gitea

Introduction: accéder à mon Localhost depuis mon Téléphone

Avant d'inviter mon équipe à me rejoindre sur Gitea pour partager nos codes sources et ressources, je me suis d'abord demandé s'il était possible d'accéder à mon **localhost** depuis mon téléphone.

La raison de ce questionnement était liée à :

"Vos codes sources de TP et projets dans les ressources de développement à partager entre tous (Les membres de votre équipe de SAÉ ou plus)."

— Mr. Carle, sae203-4

C'est là que le mode d'accès réseau NAT a attiré mon attention, notamment avec cette remarque :

"N'oubliez pas non plus de vérifier que vous avez correctement configuré la redirection de port (ce qui pourrait expliquer certains de vos soucis d'utilisation)."

— Mr. Carle, sae203-4

N'ayant pas la motivation de lire en détail le [manuel](#) de VirtualBox sur le réseau, j'ai préféré regarder des vidéos YouTube.

Une en particulier m'a bien aidé: [**Le réseau avec VirtualBox : NAT, Bridge, Host-Only, réseau interne, etc.**](#), réalisé par IT-Connect.

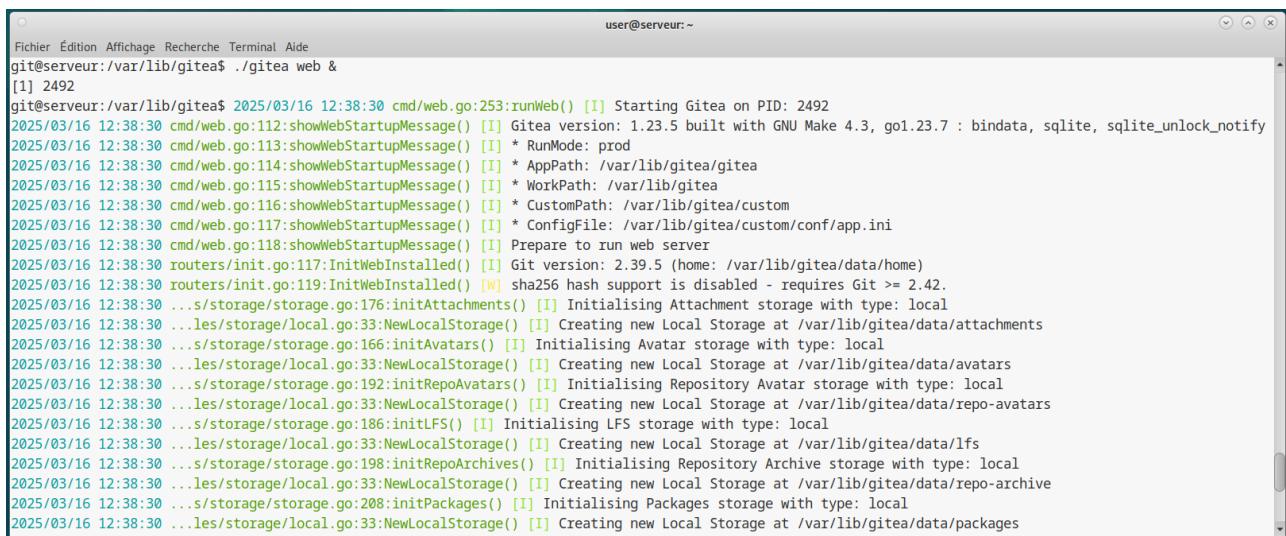
Grâce à cette vidéo, j'ai mieux compris le fonctionnement du mode d'accès réseau ainsi que du [tableau de synthèse](#) permettant de comprendre les différentes caractéristiques de ces modes d'accès réseau et une aperçu de comment résoudre mon problème initial : **accéder à la page de Gitea depuis mon téléphone**.

Tableau 1. Modes d'accès réseau

| Mode | VM → Host | VM ← Host | VM1 ↔ VM2 | VM → Net/LAN | VM ← Net/LAN |
|-------------------|-----------|---------------------|-----------|--------------|---------------------|
| NAT | ✓ | redirection de port | - | ✓ | redirection de port |
| Réseau NAT | ✓ | redirection de port | ✓ | ✓ | redirection de port |
| Accès part pont | ✓ | ✓ | ✓ | ✓ | ✓ |
| Réseau interne | - | - | ✓ | - | - |
| Réseau privé hôte | ✓ | ✓ | ✓ | - | - |

- Première tentative

Je suis allé sur ma machine virtuelle et j'ai lancé Gitea en arrière-plan avec : `./gitea web &`

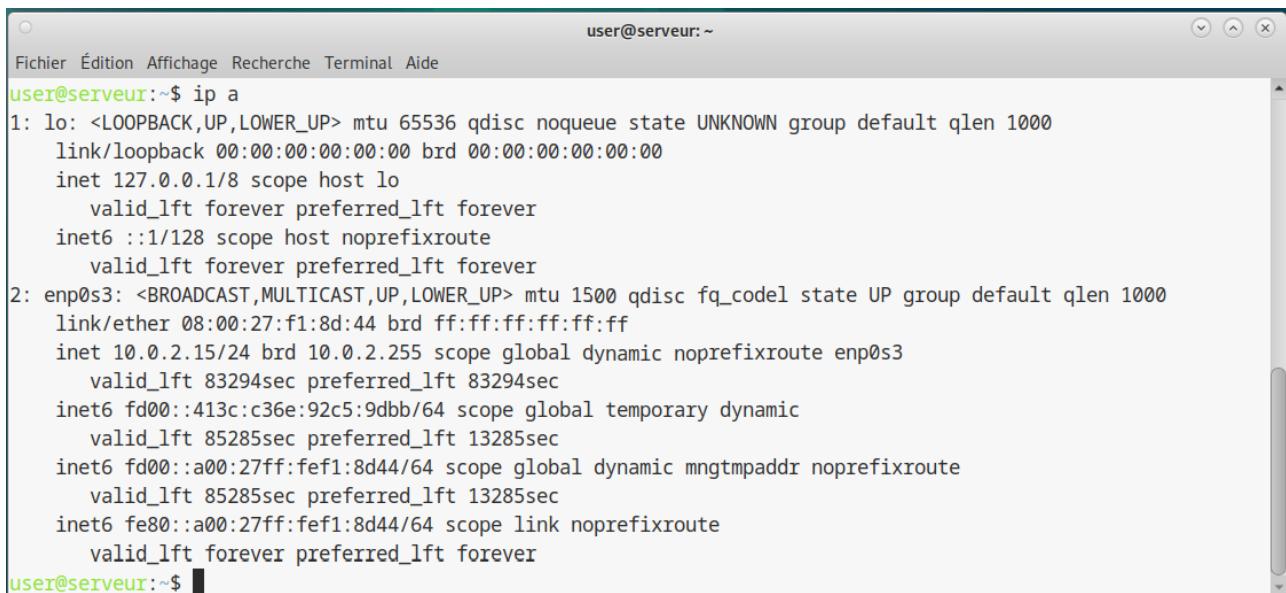


```

Fichier Édition Affichage Recherche Terminal Aide
user@serveur:~ Fichier Édition Affichage Recherche Terminal Aide
git@serveur:/var/lib/gitea$ ./gitea web &
[1] 2492
git@serveur:/var/lib/gitea$ 2025/03/16 12:38:30 cmd/web.go:253:runWeb() [I] Starting Gitea on PID: 2492
2025/03/16 12:38:30 cmd/web.go:112:showWebStartupMessage() [I] Gitea version: 1.23.5 built with GNU Make 4.3, go1.23.7 : bindata, sqlite, sqlite_unlock_notify
2025/03/16 12:38:30 cmd/web.go:113:showWebStartupMessage() [I] * RunMode: prod
2025/03/16 12:38:30 cmd/web.go:114:showWebStartupMessage() [I] * AppPath: /var/lib/gitea/gitea
2025/03/16 12:38:30 cmd/web.go:115:showWebStartupMessage() [I] * WorkPath: /var/lib/gitea
2025/03/16 12:38:30 cmd/web.go:116:showWebStartupMessage() [I] * CustomPath: /var/lib/gitea/custom
2025/03/16 12:38:30 cmd/web.go:117:showWebStartupMessage() [I] * Configfile: /var/lib/gitea/custom/conf/app.ini
2025/03/16 12:38:30 cmd/web.go:118:showWebStartupMessage() [I] Prepare to run web server
2025/03/16 12:38:30 routers/init.go:117:InitWebInstalled() [I] Git version: 2.39.5 (home: /var/lib/gitea/data/home)
2025/03/16 12:38:30 routers/init.go:119:InitWebInstalled() [W] sha256 hash support is disabled - requires Git >= 2.42.
2025/03/16 12:38:30 ...s/storage/storage.go:176:initAttachments() [I] Initialising Attachment storage with type: local
2025/03/16 12:38:30 ...les/storage/local.go:33>NewLocalStorage() [I] Creating new Local Storage at /var/lib/gitea/data/attachments
2025/03/16 12:38:30 ...s/storage/storage.go:166:initAvatars() [I] Initialising Avatar storage with type: local
2025/03/16 12:38:30 ...les/storage/local.go:33>NewLocalStorage() [I] Creating new Local Storage at /var/lib/gitea/data/avatar
2025/03/16 12:38:30 ...s/storage/storage.go:192:initRepoAvatars() [I] Initialising Repository Avatar storage with type: local
2025/03/16 12:38:30 ...les/storage/local.go:33>NewLocalStorage() [I] Creating new Local Storage at /var/lib/gitea/data/repo-avatars
2025/03/16 12:38:30 ...s/storage/storage.go:186:initLFS() [I] Initialising LFS storage with type: local
2025/03/16 12:38:30 ...les/storage/local.go:33>NewLocalStorage() [I] Creating new Local Storage at /var/lib/gitea/data/lfs
2025/03/16 12:38:30 ...s/storage/storage.go:198:initRepoArchives() [I] Initialising Repository Archive storage with type: local
2025/03/16 12:38:30 ...les/storage/local.go:33>NewLocalStorage() [I] Creating new Local Storage at /var/lib/gitea/data/repo-archive
2025/03/16 12:38:30 ...s/storage/storage.go:208:initPackages() [I] Initialising Packages storage with type: local
2025/03/16 12:38:30 ...les/storage/local.go:33>NewLocalStorage() [I] Creating new Local Storage at /var/lib/gitea/data/packages

```

Puis, j'ai utilisé : `ip a` pour lister les interfaces réseau et récupérer l'IP de `enp0s3`.

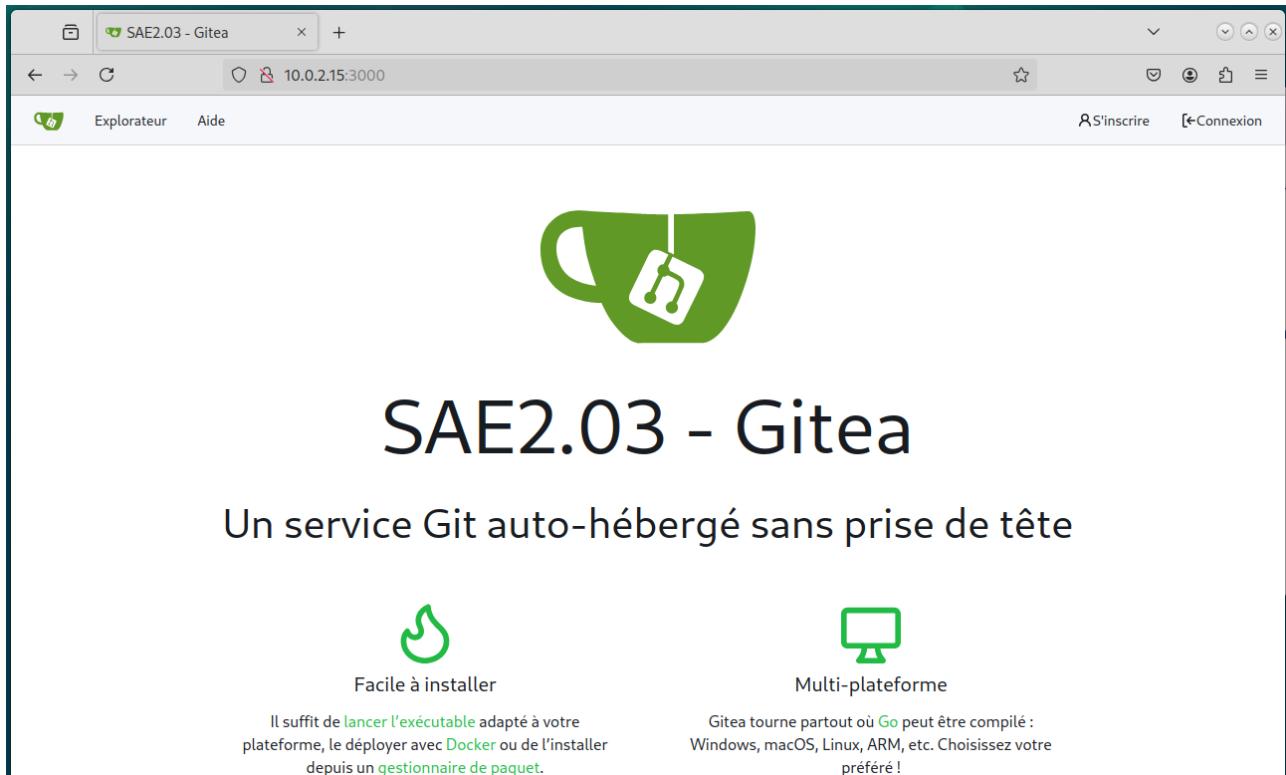


```

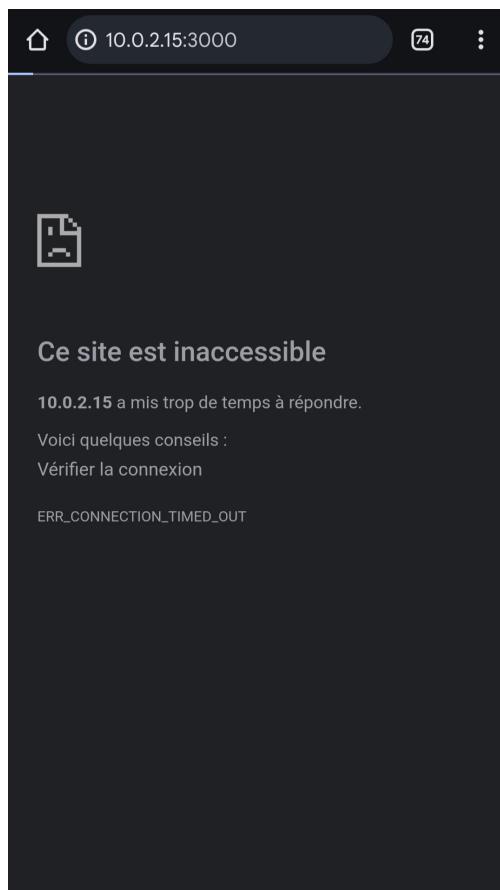
Fichier Édition Affichage Recherche Terminal Aide
user@serveur:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:f1:8d:44 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global dynamic noprefixroute enp0s3
        valid_lft 83294sec preferred_lft 83294sec
    inet6 fd00::413c:c36e:92c5:9dbb/64 scope global temporary dynamic
        valid_lft 85285sec preferred_lft 13285sec
    inet6 fd00::a00:27ff:fef1:8d44/64 scope global dynamic mngtmpaddr noprefixroute
        valid_lft 85285sec preferred_lft 13285sec
    inet6 fe80::a00:27ff:fef1:8d44/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
user@serveur:~$ 

```

J'ai testé sur Firefox avec <http://10.0.2.15:3000>, et ça fonctionnait bien sur ma VM.



Mais en essayant sur le navigateur de mon téléphone, ça ne marchait pas.



Pour vérifier si le problème venait de moi, j'ai ouvert le **cmd** sur ma machine hôte

et tenté un ping :

ping 10.0.2.15

Résultat : ça ne répondait pas... J'ai compris que ce n'était pas la bonne IP à utiliser.

```
PS C:\Users\ridhi> ping 10.0.2.15

Envoi d'une requête 'Ping' 10.0.2.15 avec 32 octets de données :
Déjà d'attente de la demande dépassé.

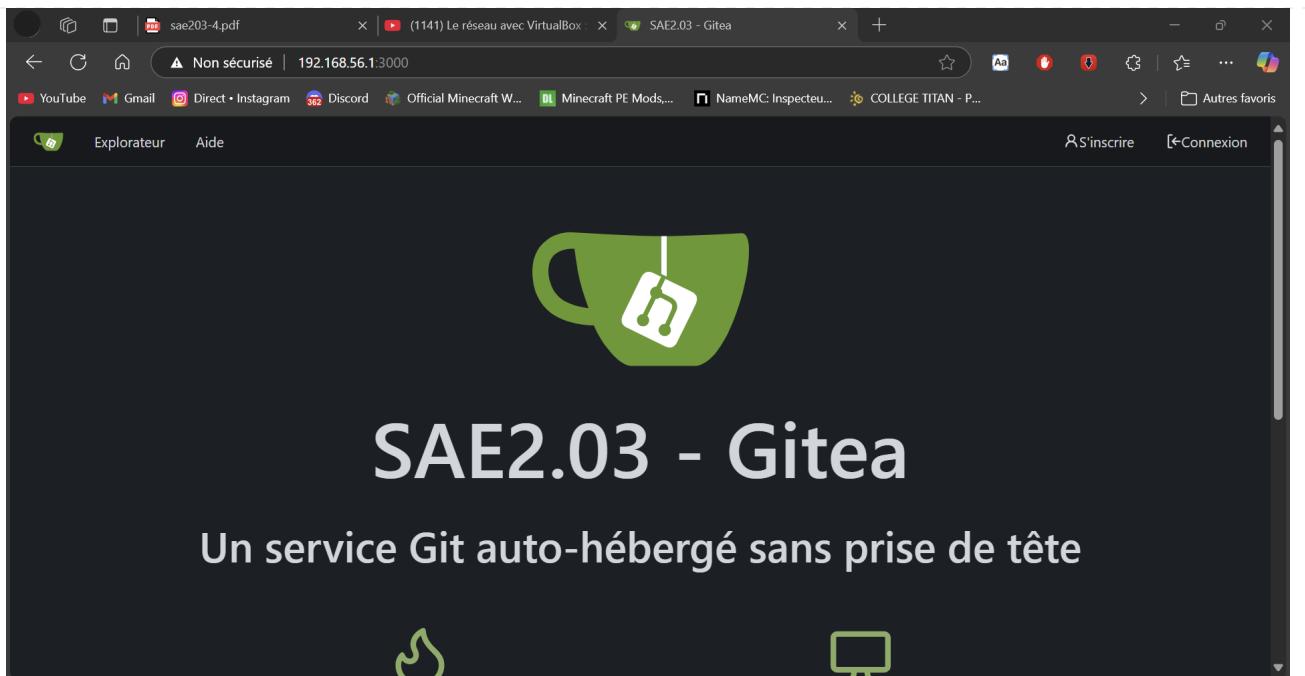
Statistiques Ping pour 10.0.2.15:
  Paquets : envoyés = 4, reçus = 0, perdus = 4 (perte 100%),
PS C:\Users\ridhi> |
```

- Deuxième tentative

J'ai laissé tomber la VM pour me concentrer sur ma machine hôte.

Toujours dans le **cmd**, j'ai utilisé : **ipconfig** pour voir mon adresse IP, et j'ai donc noté **192.168.56.1**.

J'ai tenté d'accéder à <http://192.168.56.1:3000> depuis mon navigateur PC → ça a marché.



Mais sur mon téléphone, même écran d'erreur qu'avec le précédent...

• Troisième tentative

J'ai bloqué un moment en me demandant ce que j'avais fait de travers.

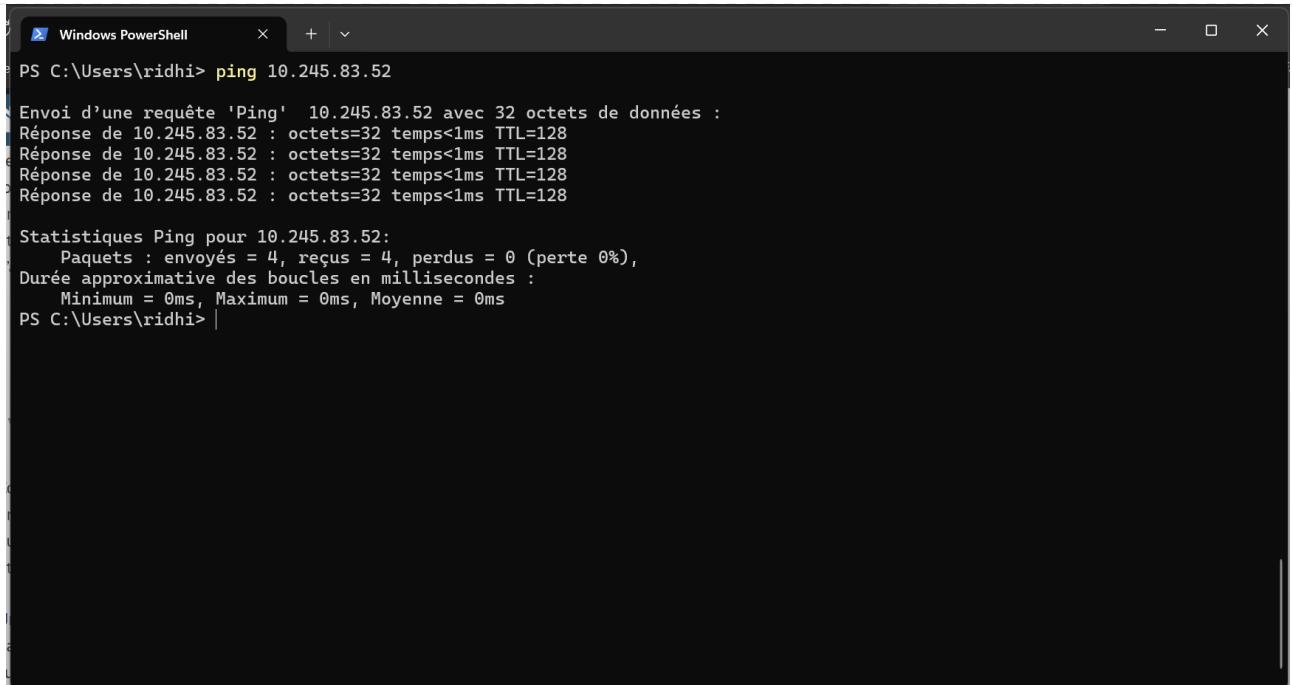
Puis, illumination!

J'aurais dû me rappeler des cours de réseau et faire la distinction entre IP publique et privée.

Je suis retourné sur cmd pour trouver mon adresse publique.

Carte réseau sans fil Wi-Fi :

J'ai tenté de la pinger : ca a marché.



```
Windows PowerShell      + | - X
PS C:\Users\ridhi> ping 10.245.83.52

Envoi d'une requête 'Ping' 10.245.83.52 avec 32 octets de données :
Réponse de 10.245.83.52 : octets=32 temps<1ms TTL=128

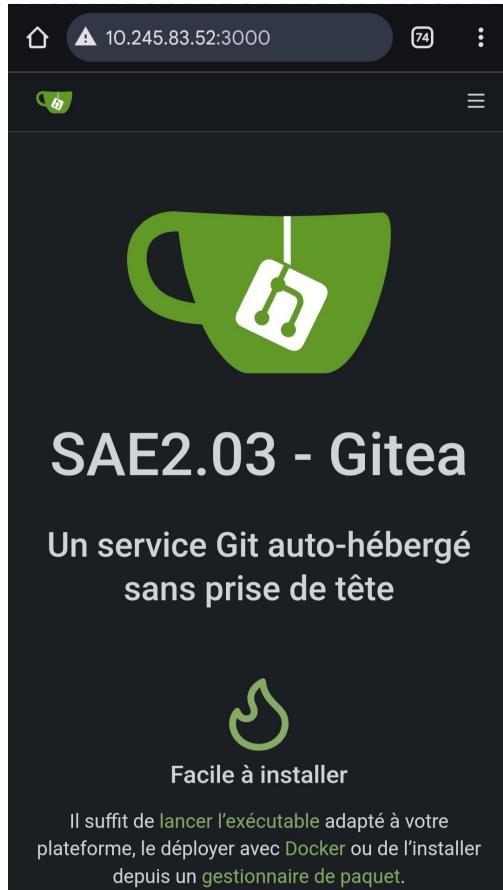
Statistiques Ping pour 10.245.83.52:
    Paquets : envoyés = 4, reçus = 4, perdus = 0 (perte 0%),
    Durée approximative des boucles en millisecondes :
        Minimum = 0ms, Maximum = 0ms, Moyenne = 0ms
PS C:\Users\ridhi>
```

J'ai essayé <http://10.245.83.52:3000> dans le navigateur de mon téléphone → ça ne marchait toujours pas.

Mais cette fois-ci, j'ai tout de suite compris le problème : le Wi-Fi.

Mon PC était connecté à **eduroam**, tandis que mon téléphone était sur le Wi-Fi de ma résidence.

Je me suis connecté à **eduroam** sur mon téléphone, retenté l'accès, et bingo, ça marchait!



- **Conclusion**

Grâce à ça, j'ai compris l'utilité des cours dans cette SAE et comment faire ce qui nous était demandé :

- Vos codes sources de TP et projets dans les ressources de développement à partager entre tous (Les membres de votre équipe de saé ou plus). Vous devez bien sûr faire des modifications et vérifier que chacune des personnes ayant accès au projet est capable de voir/récupérer le projet, voire est capable de le modifier lorsqu'elle en a les droits (ce qui implique de tester plusieurs cas de figure sur les droits des utilisateurs de vos projets sur votre serveur *Gitea*).

J'ai par la suite demandé à mon équipe de se connecter au Wi-Fi **eduroam** pour accéder à Gitea via mon adresse ip.



Note chronologique : L'introduction de cette section du rapport a été rédigée un samedi.

Partie 1: Que des problèmes mais une solution

Après avoir compris comment les autres membres de l'équipe pouvaient accéder à

mon service Gitea, j'ai laissé mon ordinateur allumé chez moi, connecté au WiFi **eduroam**.

Ensuite, je me suis rendu en salle de TP pour demander aux autres de tester l'accès en entrant mon adresse IP suivie du port 3000 : <ip>:3000.

Malheureusement, ni eux ni moi ne pouvions y accéder.

J'ai d'abord pensé que mon PC s'était éteint, alors je suis retourné vérifier.

Une fois arrivé à ma résidence, j'ai constaté que le service fonctionnait parfaitement depuis chez moi, mais qu'il était toujours inaccessible depuis l'IUT.

J'en ai alors déduit que le réseau **eduroam** du campus de **Pont de Bois** n'était probablement pas le même que celui de **Cité Scientifique**.

Avec mes connaissances actuelles, je suppose qu'il existe plusieurs sous-réseaux **eduroam** selon les campus, mais je n'ai pas approfondi la question.

- **Changement de stratégie**

J'ai alors décidé de ramener mon ordinateur sur place pour voir si, une fois connecté à eduroam de l'IUT, les autres pouvaient y accéder.

Résultat : ça fonctionnait, mais uniquement sur leurs téléphones. Les PC des salles de TP n'étaient pas connectés à eduroam, contrairement à ce que je pensais.

- **Nouvelle tentative**

Face à ce problème, nous avons décidé d'installer ma machine virtuelle sur les ordinateurs de l'IUT.

Cependant, je ne pouvais pas utiliser VirtualBox sur ma session, donc nous avons dû la mettre sur un autre compte.

J'ai transféré la VM sur ma carte mémoire, puis sur mon téléphone, avant de la brancher sur un PC de l'IUT.

Problème : nous n'avions pas les droits d'écriture sur la carte mémoire, et la seule solution connue pour contourner cela était d'avoir les privilèges sudo.

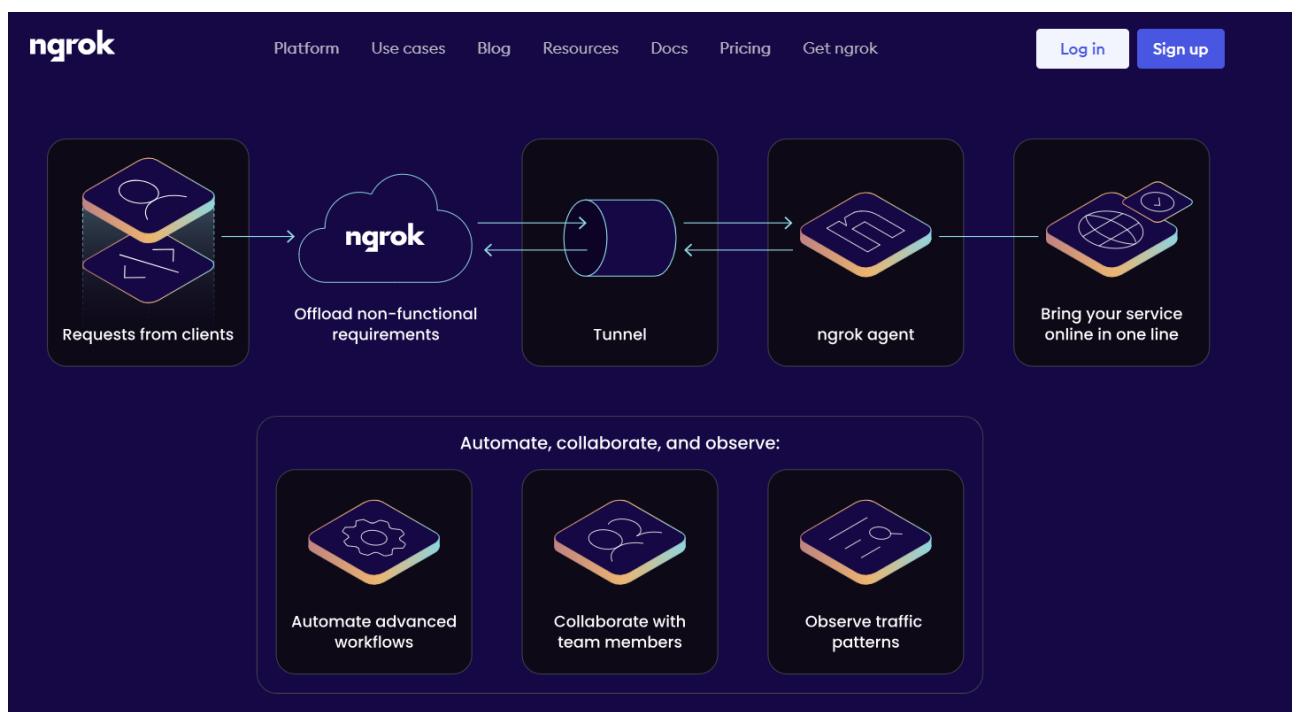
Nous avons ensuite tenté de copier la VM dans le répertoire où l'on nous avait initialement demandé de stocker nos machines virtuelles, mais la copie était extrêmement lente et ne progressait quasiment pas.

On a préféré éviter de créer une nouvelle machine virtuelle depuis zéro, par souci de **gain de temps** et parce que **nous ne sommes pas toujours à l'IUT**.

- **Solution alternative : utiliser un tunnel sécurisé** Pour ne pas perdre encore plus de temps, nous avons pris une décision radicale : mettre le service en **ligne temporairement** afin qu'il soit accessible depuis n'importe quel appareil, sans dépendre du réseau et du lieu.

Nous avons utilisé **Ngrok**, un outil permettant d'exposer un serveur local sur Internet via un tunnel sécurisé.

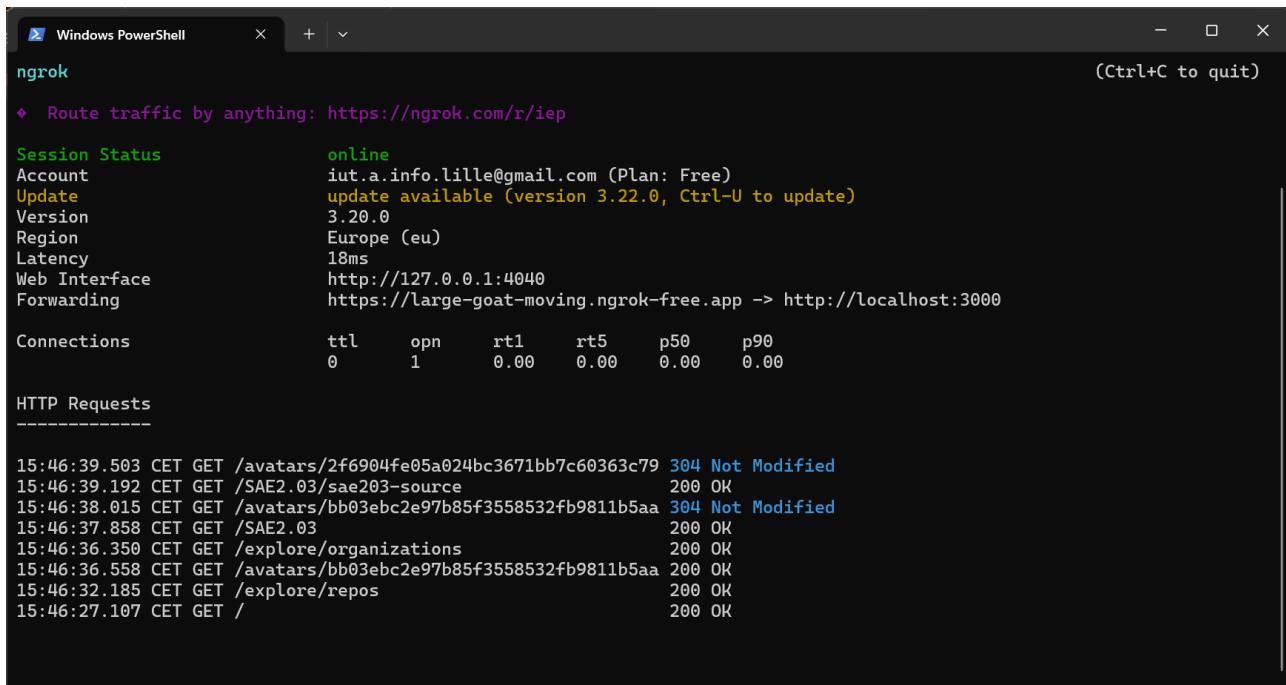
En clair, Ngrok génère une URL publique temporaire que l'on peut partager pour accéder au service local sans avoir besoin de configuration réseau complexe.



1. J'ai installé **Ngrok** sur mon PC
2. J'ai démarré ma machine virtuelle et lancé Gitea sur mon serveur local
3. J'ai exécuté `ngrok http 3000` pour générer une URL publique
4. Pour obtenir une **URL stable** qui ne change pas à chaque exécution, j'ai

enregistré un sous-domaine sur le site de Ngrok

5. Désormais, avec la commande : `ngrok http --url=large-goat-moving.ngrok-free.app 3000`, tout le monde pouvait accéder à mon serveur local en visitant l'URL `large-goat-moving.ngrok-free.app`



```
Windows PowerShell

ngrok

♦ Route traffic by anything: https://ngrok.com/r/iep

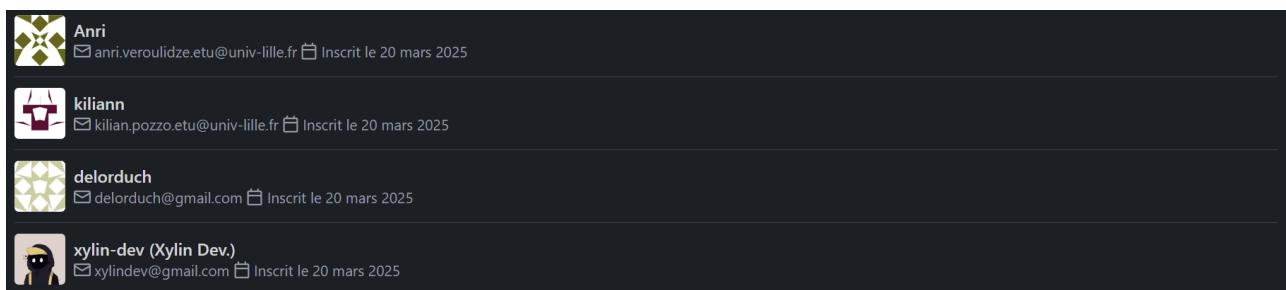
Session Status      online
Account             iut.a.info.lille@gmail.com (Plan: Free)
Update              update available (version 3.22.0, Ctrl-U to update)
Version             3.20.0
Region              Europe (eu)
Latency             18ms
Web Interface       http://127.0.0.1:4040
Forwarding          https://large-goat-moving.ngrok-free.app -> http://localhost:3000

Connections          ttl     opn     rt1     rt5     p50     p90
                      0       1      0.00    0.00    0.00    0.00

HTTP Requests
-----
15:46:39.503 CET GET /avatars/2f6904fe05a024bc3671bb7c60363c79 304 Not Modified
15:46:39.192 CET GET /SAE2.03/sae203-source 200 OK
15:46:38.015 CET GET /avatars/bb03ebc2e97b85f3558532fb9811b5aa 304 Not Modified
15:46:37.858 CET GET /SAE2.03 200 OK
15:46:36.350 CET GET /explore/organizations 200 OK
15:46:36.558 CET GET /avatars/bb03ebc2e97b85f3558532fb9811b5aa 200 OK
15:46:32.185 CET GET /explore/repos 200 OK
15:46:27.107 CET GET / 200 OK
```

Partie 2: Utilisation collaborative de Gitea

Une fois que tout le monde avait accès au service, peu importe sa localisation, chacun a créé son compte sur Gitea.



Nous avons ensuite créé une organisation nommée **SAE2.03**, dans laquelle nous avons formé une équipe regroupant tous les membres de l'équipe.

Dépôts 1 Projets 0 Paquets 0 Membres 4 Équipes 1

Paramètres

Owners

Aucune description

Les propriétaires ont un accès complet à tous les dépôts et disposent d'un accès administrateur de l'organisation.

Paramètres

4 Membres 1 dépôts

Chercher des utilisateurs... Ajouter un Membre

Anri
delorduch
kiliann
xylin-dev (Xylin Dev.)

Exclude
Exclude
Exclude
Exclude

Puis, nous avons mis en place un nouveau dépôt nommé **sae203-source**.

SAE2.03 / sae203-source

Code Tickets Demandes d'ajout Actions Paquets Projets Publications Wiki Activité Paramètres

34 Révisions 1 Branche 0 Étiquette

Chercher du code... Q

Description
Aucune description fournie

Gérer les sujets

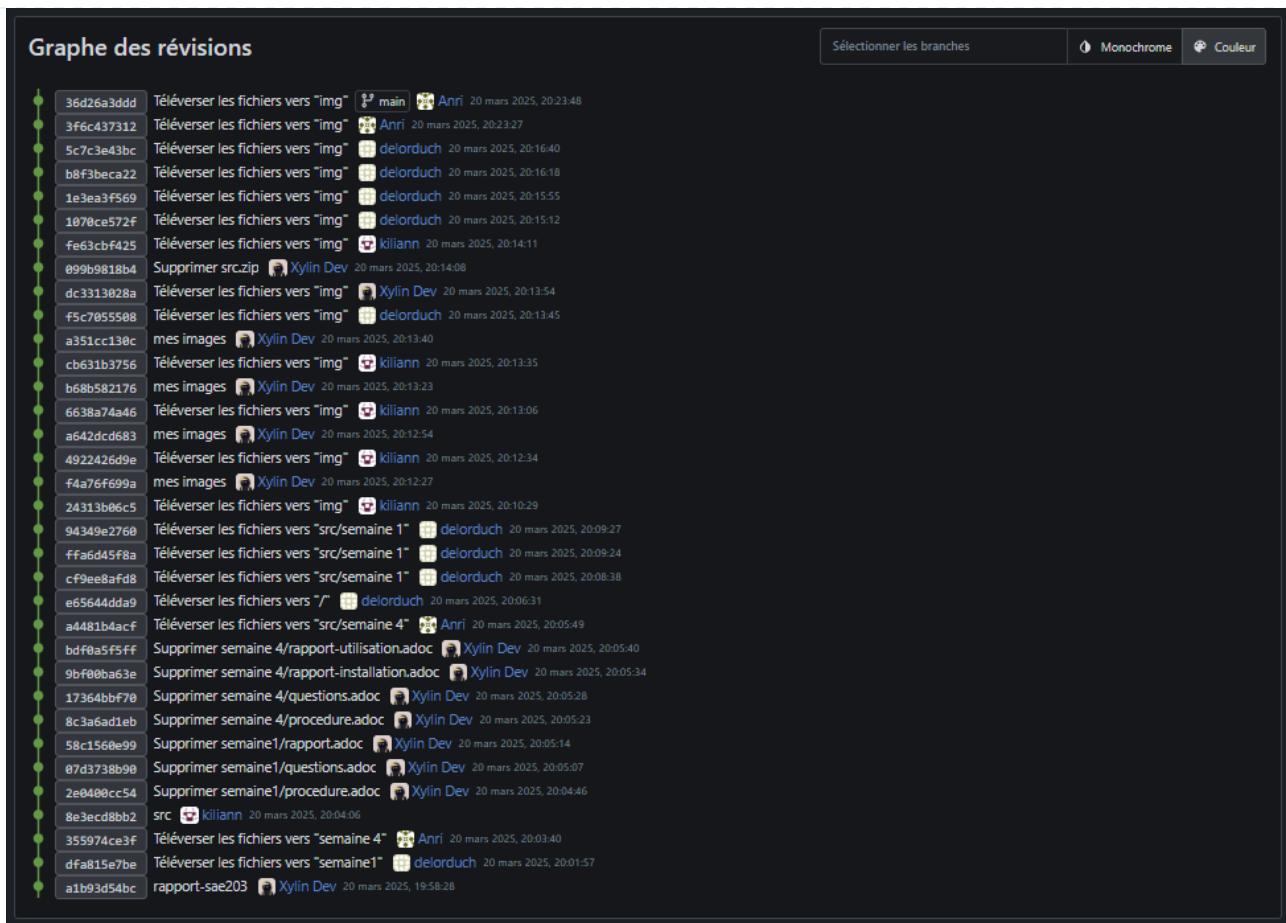
Lisez-moi

12 MiB

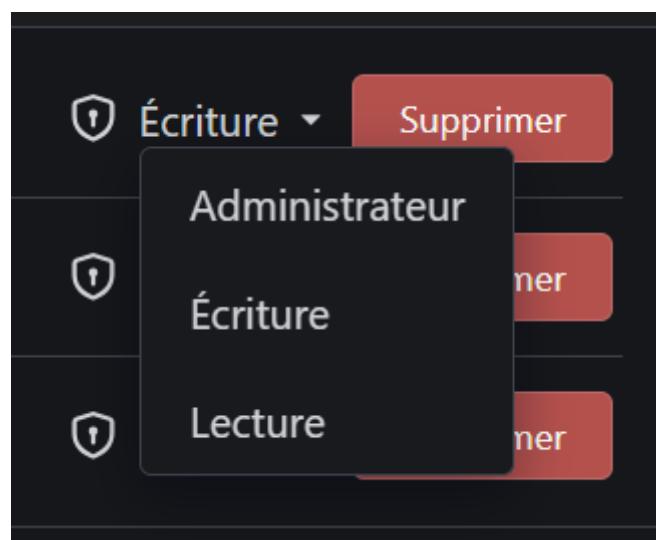
Langue
HTML 100%

| Fichier | Contenu | Date |
|--------------|--|------------|
| archive | rapport-sae203 | avant-hier |
| font | rapport-sae203 | avant-hier |
| img | Téléverser les fichiers vers "img" | avant-hier |
| src | Téléverser les fichiers vers "src/semaine 1" | avant-hier |
| theme | rapport-sae203 | avant-hier |
| rapport.adoc | rapport-sae203 | avant-hier |
| rapport.html | rapport-sae203 | avant-hier |
| rapport.pdf | rapport-sae203 | avant-hier |
| readme.adoc | rapport-sae203 | avant-hier |

Où chacun d'entre nous a push/téléversé sa partie du rapport.



Dans ce repository, nous n'avons pas jugé utile de configurer des permissions spécifiques, car il était exclusivement destiné à notre équipe et les consignes ne semblaient pas l'exiger à ce stade.



Cependant, nous avons, par la suite, décidé de créer une autre équipe avec des **permissions de lecture** ou **aucun accès**, au cas où les membres du groupe E souhaiteraient jeter un œil et tester notre service Gitea.

| Permettre l'accès aux Sections du dépôt * | | | |
|--|---------------|-----------|------------|
| Unité | Aucun accès ⓘ | Lecture ⓘ | Écriture ⓘ |
| Code Accéder au code source, fichiers, révisions et branches. | ● | ○ | ● |
| Tickets Organiser les rapports de bug, les tâches et les jalons. | ○ | ● | ● |
| Demandes d'ajout Active les demandes d'ajouts et l'évaluation du code. | ● | ○ | ● |
| Publications Suivi des publications et des téléchargements. | ○ | ● | ● |
| Wiki Écrire et partager de la documentation avec vos collaborateurs. | ○ | ● | ● |
| Projets Gérer les tickets et les demandes d'ajouts dans les projets. | ○ | ● | ● |
| Paquets Gérer les paquets du dépôt. | ○ | ● | ● |
| Actions Gérer les actions | ○ | ● | ● |

Dans cette équipe, nous avons, pour l'instant, uniquement ajouté les utilisateurs concernés par la prochaine étape des consignes, afin de limiter l'accès aux personnes réellement impliquées.

Ensuite, nous avons chacun repris sa **SAE1.02** (*concernant le logiciel ludopédagogique*), faute d'accès aux fichiers de nos TP à ce moment-là, et nous avons chacun créé son propre repository afin d'y **push/pull** nos fichiers ou de les **téléverser directement** via l'interface web.

| | |
|---|--|
|  Anri / SAE_Ludo-pedagogique Actualisé avant-hier |  Java ⭐ 0 ⚡ 0 |
|  delorduch / Ludo-pedagogique <small>Privé</small> Ce depot contient le Jeu ludo fait en S1 Actualisé avant-hier |  Java ⭐ 0 ⚡ 0 |
|  kiliann / memory-match jeux de memoire en java Actualisé avant-hier |  Java ⭐ 0 ⚡ 0 |
|  Anri / sae203-perso Actualisé avant-hier |  AsciiDoc ⭐ 0 ⚡ 0 |
|  xylin-dev / VivreOuSurvivre Actualisé avant-hier |  Java ⭐ 0 ⚡ 0 |

Nous avons aussi invité d'autres personnes ayant participé à nos projets de SAE pour compléter nos dépôts.

Nous avons également configuré les permissions :

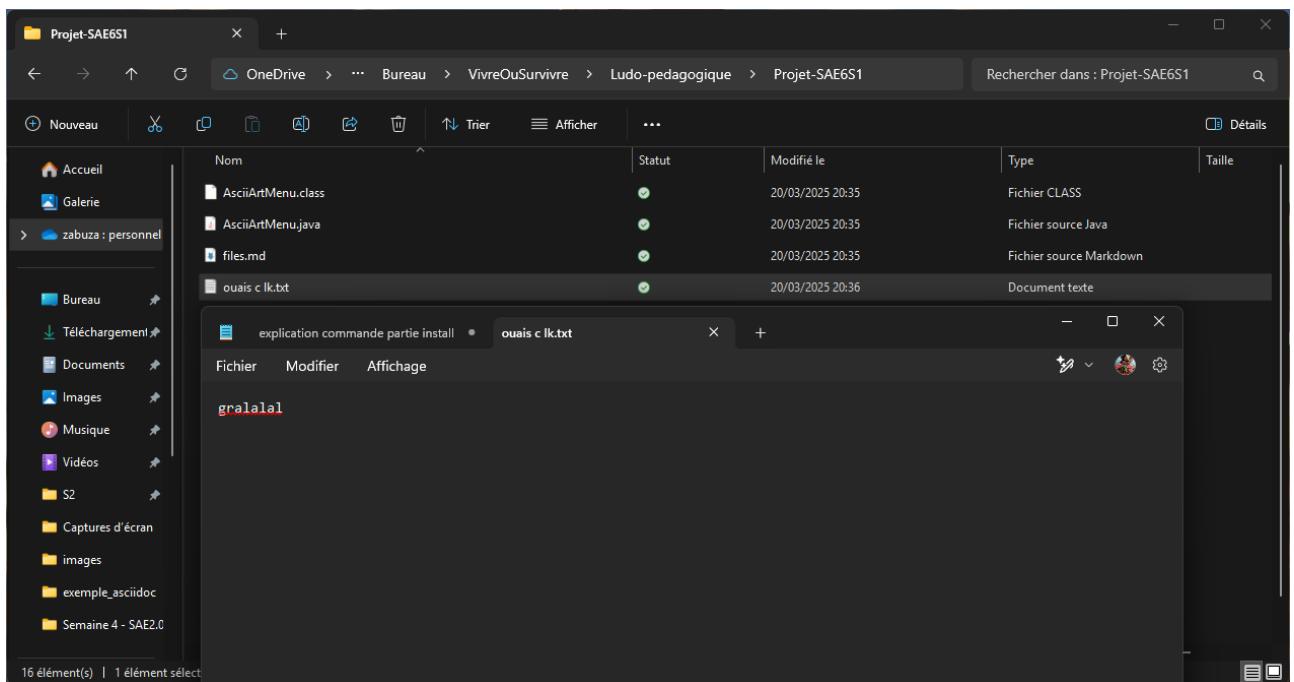
- **Administration** : réservée au propriétaire du dépôt.
- **Écriture** : attribuée aux membres ayant travaillé avec nous sur la SAE, leur permettant de modifier les dépôts.
- **Lecture** : pour ceux qui devaient uniquement consulter les fichiers, sans possibilité de modification.

Collaborateurs

| | | |
|-----------|----------|-----------|
| Anri | Lecture | Supprimer |
| kiliann | Lecture | Supprimer |
| delorduch | Lecture | Supprimer |
| SleoNiiX | Écriture | Supprimer |

Chercher des utilisateurs... Ajouter un collaborateur

Enfin, nous avons réalisé des tests afin de vérifier s'il était possible de **push** dans des repositories où nous n'avions pas les droits nécessaires.



```
xylin@Book3-360xylin:/mnt/ 
hint: git submodule add <url> Ludo-pedagogique
hint:
hint: If you added this path by mistake, you can remove it from the
hint: index with:
hint:
hint: git rm --cached Ludo-pedagogique
hint:
hint: See "git help submodule" for more information.
xylin@Book3-360xylin:/mnt/c/Users/ridhi/OneDrive/Bureau/VivreOuSurvivre$ cd Ludo-pedagogique/
xylin@Book3-360xylin:/mnt/c/Users/ridhi/OneDrive/Bureau/VivreOuSurvivre/Ludo-pedagogique$ git add .
xylin@Book3-360xylin:/mnt/c/Users/ridhi/OneDrive/Bureau/VivreOuSurvivre/Ludo-pedagogique$ git commit -m "toto est là"
[main 9d3c691] toto est là
 1 file changed, 1 insertion(+)
 create mode 100644 Projet-SAE6S1/ouais c lk.txt
xylin@Book3-360xylin:/mnt/c/Users/ridhi/OneDrive/Bureau/VivreOuSurvivre/Ludo-pedagogique$ git push
Username for 'https://large-goat-moving.ngrok-free.app': xylin-dev
Password for 'https://xylin-dev@large-goat-moving.ngrok-free.app':
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 16 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (4/4), 333 bytes | 6.00 KiB/s, done.
Total 4 (delta 1), reused 0 (delta 0), pack-reused 0
remote:
remote: Gitea: User permission denied for writing.
To https://large-goat-moving.ngrok-free.app/delorduch/Ludo-pedagogique.git
 ! [remote rejected] main -> main (pre-receive hook declined)
error: failed to push some refs to 'https://large-goat-moving.ngrok-free.app/delorduch/Ludo-pedagogique.git'
xylin@Book3-360xylin:/mnt/c/Users/ridhi/OneDrive/Bureau/VivreOuSurvivre/Ludo-pedagogique$
```

Option pour téléverser inexistant

Pour finir, nous avons utilisé Gitea comme nous utilisons habituellement GitHub ou GitLab, en testant les différentes fonctionnalités et en nous assurant de bien comprendre son fonctionnement.

4.3: Procédure pour l'installation de Gitea



Cette section vous guidera à travers les étapes nécessaires pour installer **Gitea version 1.23.5** et le configurer en tant que service sur une machine Linux.

Prérequis pour l'installation de Gitea

Avant de procéder à l'installation de Gitea sur une machine Debian, assurez-vous de remplir les conditions suivantes :

1. Accès root ou utilisateur avec privilèges sudo

Vous devez avoir un accès administrateur (*root ou utilisateur avec des privilèges sudo*) sur votre système pour installer et configurer Gitea.

2. Machine Debian 64-bit avec les paquets nécessaires

Vérifiez que votre système Debian dispose des paquets suivants : **git**, **sqlite3** et **curl**.

Si ces paquets ne sont pas installés, vous pouvez les ajouter avec la commande suivante : **sudo apt update && sudo apt upgrade && sudo apt install -y git**

`sqlite3 curl` ou si vous avez besoin d'installer Debian, référez-vous à la section dédiée : [Procédure pour l'Auto-Installation de Debian](#)

3. Configuration réseau pour la machine virtuelle

Gitea utilise par défaut le port 3000. Si vous installez Gitea sur une machine virtuelle sous VirtualBox, vous devez rediriger ce port pour pouvoir y accéder depuis votre machine hôte.

- a. Ouvrez **VirtualBox**.
- b. Sélectionnez votre machine virtuelle et cliquez sur **Configuration**.
- c. Allez dans Réseau et vérifiez que le mode d'accès réseau est réglé sur **NAT**.
- d. Cliquez sur **Redirection de ports**.
- e. Ajoutez une nouvelle règle en cliquant sur l'icône 
- f. Vous devez remplir les champs afin d'obtenir un résultat similaire à celui présenté ci-dessous puis cliquez sur **OK** et démarrez votre machine virtuelle:

| Nom | Protocol | IP hôte | Port hôte | IP invité | Port invité |
|-------|----------|---------|-----------|-----------|-------------|
| gitea | TCP | | 3000 | | 3000 |

Téléchargement des fichiers binaires

Nous allons maintenant procéder au téléchargement du fichier binaire de la version 1.23.5 et, si nécessaire, à la vérification de sa signature GPG.

Téléchargement du fichier binaire

- Ouvrez votre Terminal avec **Ctrl+Alt+T** et déplacez-vous dans le répertoire des téléchargements en utilisant la commande suivante : `cd ~/Téléchargements`.
- Exécutez la commande suivante pour télécharger Gitea 1.23.5 et le spécifier son nom en gitea :

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

- Une fois le téléchargement terminé, donnez la permission d'exécution sur ce fichier avec la commande : `chmod +x gitea`.

Vérification de la signature GPG Gitea signe tous ses binaires avec une clé GPG pour garantir qu'ils n'ont pas été modifiés de manière indésirable. Pour vérifier l'intégrité du binaire téléchargé, suivez les étapes suivantes :

- **Téléchargez le fichier de signature**

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
```

- **Récupérez la clé publique**

Utilisez la commande suivante pour récupérer la clé publique (*ID* : **7C9E68152594688862D62AF62D9AE806EC1592E2**) depuis le serveur de clés `keys.openpgp.org` :

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

- **Vérifiez la signature du fichier**

Exécutez la commande suivante pour vérifier que le fichier **gitea** n'a pas été altéré en le comparant avec la signature stockée dans le fichier **gitea-asc**:

```
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.

Préparation de l'environnement pour Gitea

La préparation de l'environnement pour Gitea est une étape importante pour assurer son bon fonctionnement.

- **Vérification de la version de Git**

Vérifiez que votre version de Git soit **supérieure ou égale à la version 2.0**, ce qui est requis pour utiliser Gitea. Utilisez la commande suivante : `git -v`

- **Création de l'utilisateur pour Gitea**

Créez un utilisateur qui exécutera Gitea (nommé `git`) avec la commande suivante :

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

- **Création des répertoires nécessaires pour Gitea**

Créez les répertoires nécessaires à l'exécution de Gitea :

```
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
```

- **Déplacement du fichier binaire de Gitea**

Déplacez le fichier binaire de `gitea`, téléchargé dans le répertoire des téléchargements, vers le répertoire `/var/lib/gitea` : `mv gitea /var/lib/gitea`

- **Démarrage manuel de Gitea**

- Changez d'utilisateur dans la ligne de commande en exécutant : `sudo su - git`.
- Puis, déplacez-vous dans le répertoire où se trouve le fichier binaire de Gitea avec la commande suivante : `cd /var/lib/gitea`.

- Enfin, exédez la commande suivante pour lancer le serveur Gitea en mode web : `./gitea web`.

Configuration et Installation de Gitea

Dernière étape : Configurer et installer le service Gitea.

Que ce soit sur votre machine hôte ou invitée, ouvrez un navigateur et suivez les étapes de configuration indiquées ci-dessous.

- Paramètres de la base de données

| | |
|-------------------------|------------------------------|
| Type de base de données | SQLite3 |
| Emplacement | /var/lib/gitea/data/gitea.db |

- Configuration générale

| | |
|--|---|
| Titre du site | SAE2.03 - Gitea (<i>ou ce que vous voulez</i>) |
| 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 |
| | <input type="checkbox"/> Activer la vérification des mises-à-jour |

- Paramètres facultatifs

- Paramètres de Messagerie
 - ignoré
- Paramètres Serveur et Tierce Parties
 - ignoré

- Paramètres de compte administrateur

| | |
|---|---------------|
| Nom d'utilisateur administrateur | gitea |
| Courriel | git@localhost |
| Mot de passe | gitea |
| Confirmez le mot de passe | gitea |

- Cliquez sur **Installer Gitea**.

L'installation sera effectuée et vous pourrez suivre son avancement directement depuis le terminal où vous avez exécuté la commande **./gitea web**.

Une fois l'installation terminée, il est recommandé de définir les permissions en lecture seule en exécutant **exit** puis en utilisant la commande suivante :

```
sudo chmod 750 /etc/gitea
sudo chmod 640 /var/lib/gitea/custom/conf/app.ini
```

Profitez pleinement du service en l'exécutant en arrière-plan, tout en vous assurant que la machine invitée reste ouverte : **./gitea web &**.

[1] Configuration du réseau - Debian

[2] MATE Desktop

[3] GNOME

[4] Serveur Web - Wikipédia

[5] Définition SSH - Oracle

[6] Proxy - Wikipédia

[7] UNAME Manuel - Debian

[8] Guest Additions - VirtualBox

[9] Mount Manuel - Debian

[10] Qu'est-ce que Debian?

[11] Comment prononce-t-on Debian et quel est le sens de ce mot ?

[12] DebianReleases

[13] Debian Long Term Support

[14] Extended Long Term Support

[15] Debian LifeSpan FAQ

[16] Debian Version

[17] Source Code Name

[18] Debian 11 Architectures

-
- [19] Git - gitk Documentation
 - [20] Git - git-gui Documentation
 - [21] What is Gitea?
 - [22] Wikipédia: Fork (développement logiciel)
 - [23] What is Gitea?
 - [24] Gogs - Un service git auto-hébergé sans-douleur.
 - [25] Global Options - Gitea Command Line
 - [26] Qu'est-ce que l'intégration continue (CI) ?
 - [27] Qu'est-ce que la livraison continue (CD) ?
 - [28] Que sont les fichiers au format ASC et ASCII ?
 - [29] Gitea Command Line - Web