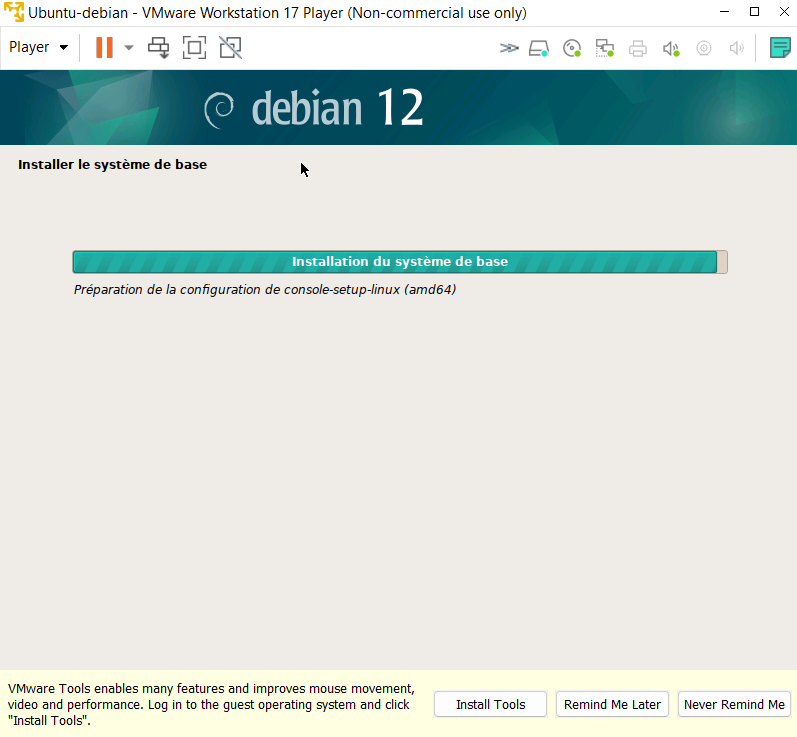
22/03/2024

**Système, Scripts et Sécurité**

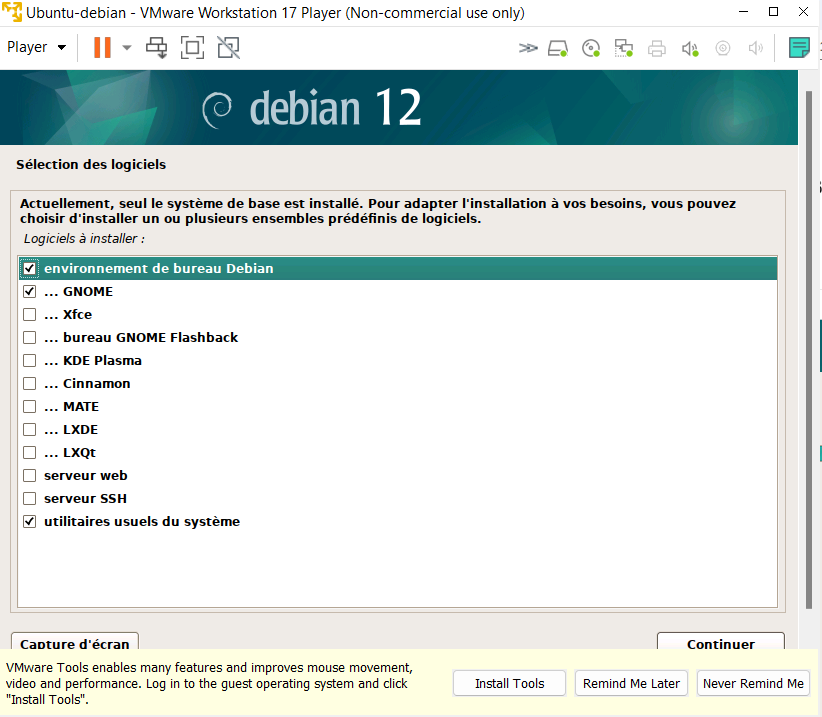
Jordan REINALDO, Lucas SAVIOZ, Manon RITTLING

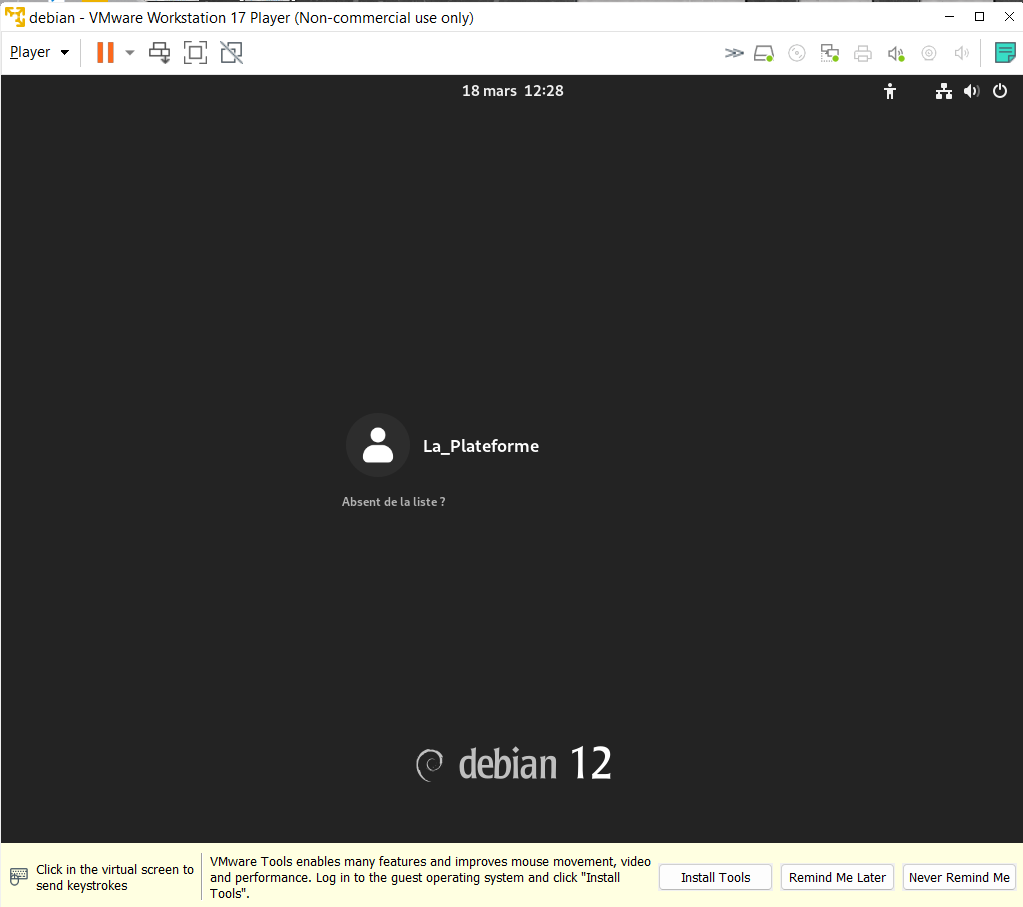
1. Création d’une VM Debian

Pour créer un VM Debian, nous avons utilisé VMware / VirtualBox. Et avons installé le système d'exploitation Debian 12.1 avec un fichier iso



Environnement graphique choisis :



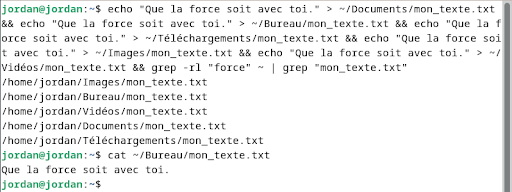


2- Commandes de recherche avancée

Pour créer un fichier txt et l’éditer, nous avons plusieurs possibilités nous en avons choisis 2

**Option 1 :**

Nous avons choisi la commande « echo » pour insérer **le texte** puis ajouter le **chemin du répertoire choisi/ le nom du fichier.txt.**



Le **&&** permet de faire plusieurs commandes en même temps

**Option 2 :**

Pour cette deuxième méthode nous avons choisi d’utiliser «**nano** » qui permet d’éditer et si le fichier n’est pas créé il le crée aussi.



Voici la commande pour copier notre fichier txt dans plusieurs dossiers



Cette commande copie le fichier **mon\_texte.txt** dans chacun des répertoires **Bureau**, **Documents**, **Images**, **téléchargements** et **vidéos**. La boucle itère sur chaque répertoire de la liste, et à chaque itération, copie le fichier **mon\_texte.txt** dans le répertoire actuel. Cette commande est utile pour copier un fichier dans plusieurs répertoires sans avoir à le faire manuellement pour chaque répertoire.

3- Compression et décompression de fichiers

Créez un répertoire nommé "Plateforme" dans le dossier "Documents" de votre session et ajoutez-y le fichier “mon\_texte.txt” précédemment créé.

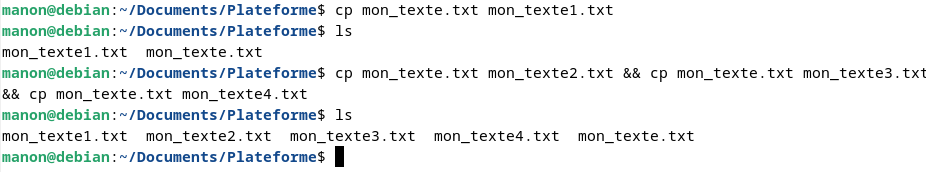
Pour crée un répertoire nommé « Plateforme » dans le dossier Documents.

**cd (change directory)**est utilisée dans le terminal pour changer le répertoire courant (c'est-à-dire le répertoire dans lequel se trouve l'utilisateur actuellement).



La commande « **cp** » sert pour copié des fichiers et répertoire d’un emplacement a un autre.

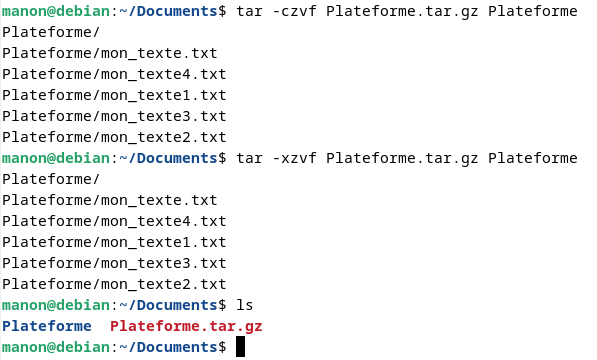
Il peut être intéressant d’utilisé « **cp -r » permet de copié le répertoire et son contenu.**



Ensuite, archivez le répertoire "Plateforme" en utilisant les commandes "tar" et

"gzip".

Que signifie **czvf** quand on fait la commande « tar »  
-**c** : Crée une nouvelle archive.  
-**z** : Filtre l'archive à travers gzip, pour la compression.  
-**v** : Mode verbeux, affiche le processus à l'écran.  
-**f** : Spécifie le nom du fichier d'archive.



**-x** : cette option indique à **tar** d'extraire les fichiers de l'archive.

4- Manipulation de texte

**Option 1 :**

Nous avons choisi d’utiliser Visual code pour crée un script python permettant la création d’un fichier

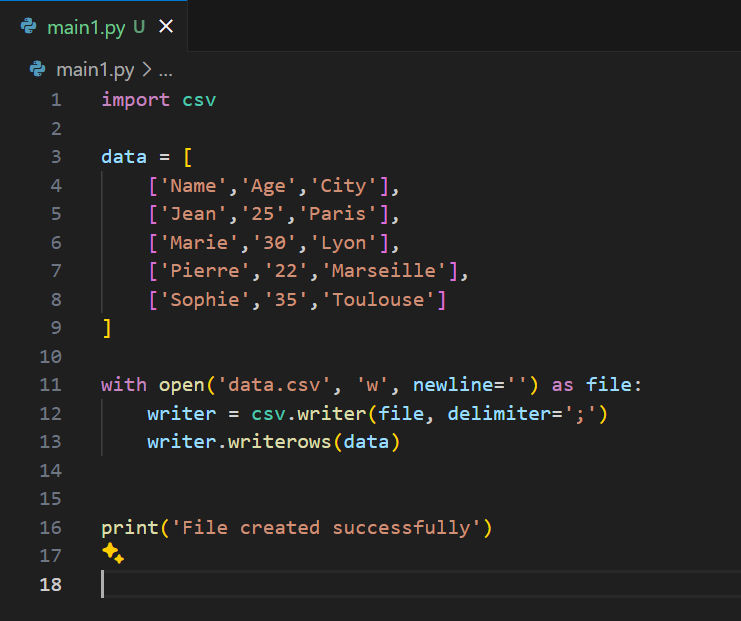
CSV et l’ajout des données suivantes :

➔ Jean, 25 ans, Paris

➔ Marie, 30 ans, Lyon

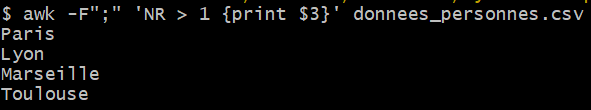
➔ Pierre, 22 ans, Marseille

➔ Sophie, 35 ans, Toulouse



Le **choix du séparateur de colonnes** dans un fichier CSV dépend de **la convention utilisée dans un pays donné**. Par exemple, le «**;** » est couramment utilisé en France, tandis que la « **,** » est couramment utilisée aux États-Unis.

Et fait cette commande dans le terminal :



* **awk** : cette commande est utilisée pour traiter des fichiers texte en utilisant des expressions régulières et des commandes de programmation.
* **-F ";"** : cette option indique à **awk** d'utiliser le point-virgule comme séparateur de colonnes dans le fichier CSV.
* **'NR > 1 {print $3}'** : cette option indique à **awk** d'exécuter la commande **{print $3}** pour chaque ligne du fichier, sauf pour la première ligne (**NR > 1** signifie "nombre de ligne supérieur à 1"). La commande **{print $3}** affiche la troisième colonne de chaque ligne.
* **data.csv** : c'est le nom du fichier CSV à traiter.

**Option 2 :**

Tout d’abord installer les paquets, il est **important de passer en root** pour installer les paquets

Paquet vim (éditeur de texte que l’on peut utiliser directement depuis le terminal



Apres nous avons installé le paquet « **virtualenv** » qui permet de crée un nouvel environnement

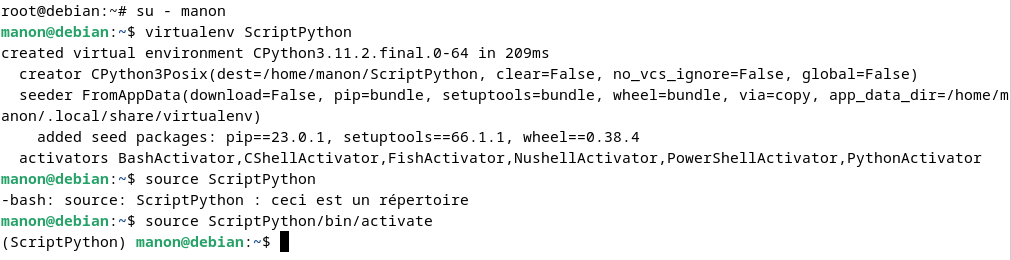
**Créer un nouvel environnement** sur Linux offre une manière pratique et flexible de gérer les dépendances logicielles, assurant ainsi :

* **Isolation des dépendances** : Éviter les conflits entre différentes versions de bibliothèques ou de paquets logiciels, garantissant le bon fonctionnement de l'application.
* **Développement et test efficaces** : Faciliter la gestion des dépendances pour chaque projet, assurant une configuration cohérente pour toute l'équipe de développement.
* **Gestion du versionnement**: Permettre de gérer différentes versions de dépendances pour des projets distincts, sans provoquer de conflits.
* **Renforcement de la sécurité** : Limiter l'accès des applications à certaines ressources système, offrant une couche de protection supplémentaire, notamment pour les applications tierces.
* **Organisation et nettoyage** : Maintenir le système propre et bien organisé en séparant les environnements pour différents projets ou applications, facilitant ainsi la désinstallation des dépendances spécifiques lorsqu'elles ne sont plus nécessaires.

En somme, la création d'un nouvel environnement sur Linux assure une gestion efficace des dépendances logicielles, garantit la cohérence et l'isolation entre les projets, et contribue à une meilleure gestion des ressources système.



Après nous pouvons donc crée notre nouvel environnement comme ceci :



Permet de vérifier si l’environnement a bien été créé d’activer notre nouvel environnement

Dans le nouvel environnement appelé « **ScriptPython** » on installe le paquet « **psutil** »

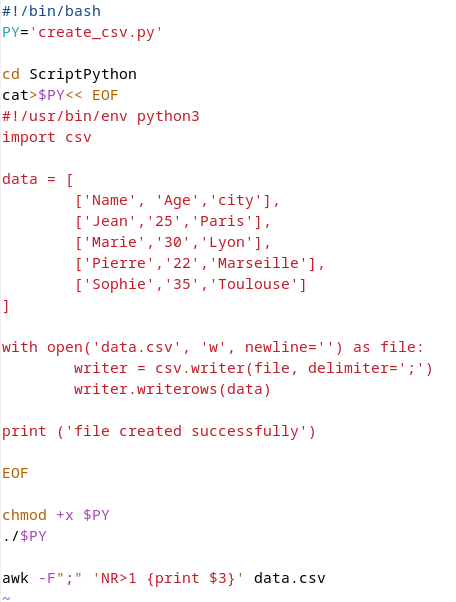
**PSUTIL c’est quoi :**

**psutil** est une **bibliothèque Python** qui permet d'interagir avec les processus et d'obtenir des **informations sur les ressources système** telles que la mémoire, le CPU, le disque et le réseau. Elle est utile pour surveiller et gérer les processus en cours d'exécution, diagnostiquer les problèmes de performance, détecter les goulets d'étranglement, optimiser l'utilisation des ressources et surveiller le trafic réseau. En résumé, psutil est un outil précieux pour les administrateurs système, les développeurs d'applications et les ingénieurs de performance.

Apres nous tapons cette commande pour crée notre script python :



Voici notre script pour créer le fichier data avec les données dedans, le convertir en csv et récupérer la colonne 3 du fichier qui contient les **villes**



Ce raccourci permet de créer un fichier Python nommé **$PY** avec le contenu fourni. Le contenu est encapsulé entre **<< EOF** et **EOF**, permettant d'écrire plusieurs lignes de texte sans avoir à les écrire une par une dans le fichier.

Ce raccourci est utile lorsque vous avez besoin de créer rapidement un script ou un fichier avec un contenu prédéfini sans avoir à ouvrir un éditeur de texte.

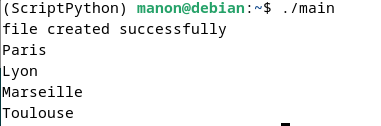
Exemple de script rapide directement en ligne de commande



Pour pouvoir exécuter son fichier script, il faut lui donner les droits d’exécution avec la commande suivante :

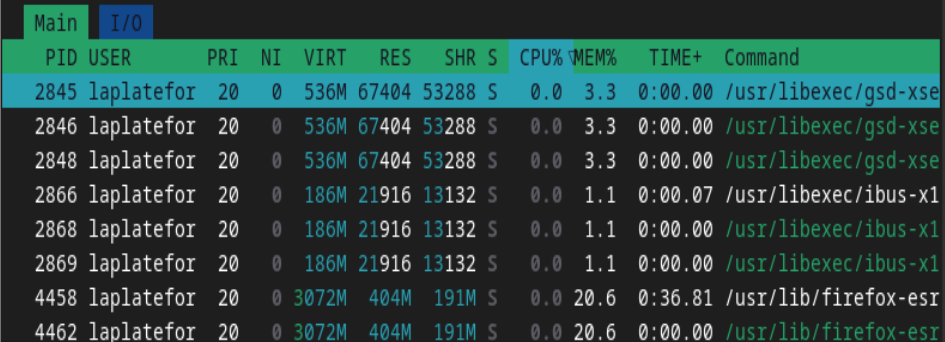


Commande pour exécuter le script



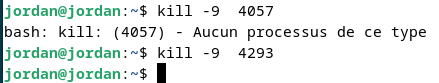
5- Gestion des processus

Pour cela on récupère tous les processus du système on utilise la commande "**ps aux**",   
cela permet donc de visualiser caractéristiques et d'analyser les données.

  
  
De préférence et par question d'efficacité on utilise « **top**» ou «**htop** » qui est **exécuter en temps réel,** afin de récupérer directement les données changeantes lors de l'utilisation de firefox.  
  
On utilise la commande "**pgrep firefox**" afin de cibler le programme voulu.

  
  
On peut donc prendre par exemple le navigateur Firefox et récupérer le PID pour arrêter son processus.

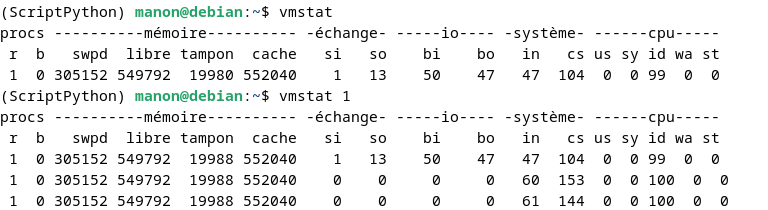
* **kill + pid** : cette commande envoie le signal TERM (terminate) au processus avec l'ID spécifié par **pid**. Le signal TERM demande au processus de se terminer proprement en effectuant toutes les tâches de nettoyage nécessaires avant de se terminer.
* **kill -9 + pid** : cette commande envoie le signal KILL (kill) au processus avec l'ID spécifié par **pid**. Le signal KILL demande au processus de se terminer immédiatement, sans effectuer aucune tâche de nettoyage. Cette commande doit être utilisée avec précaution, car elle peut entraîner une perte de données ou des problèmes de stabilité du système si le processus est en train d'effectuer des opérations critiques.



6- Surveillance des ressources système

Surveillance en temps réel de l’utilisation du CPU, de la  
mémoire et d’autres ressources système

**Vmstat** permet de récupérer une capture l’état présent de la surveillance alors que **vmstat 1** permet une visualisation en temps réel et continue de l’état des ressources du système

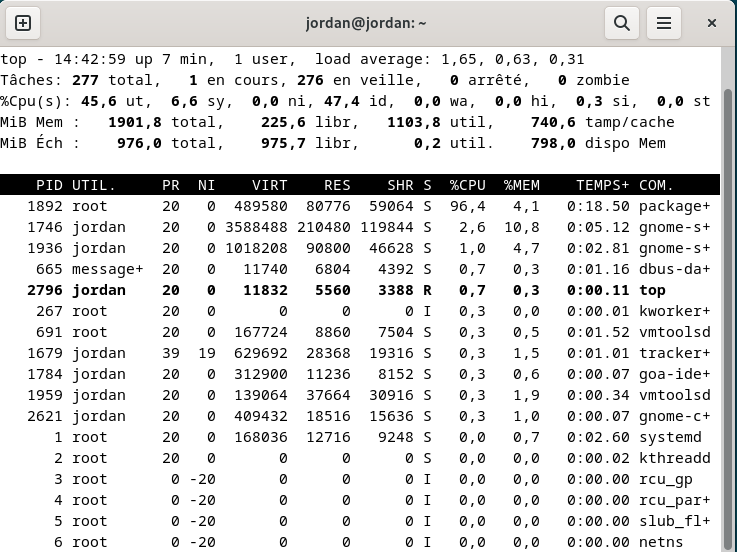


La commande **top** permet de surveiller les processus en temps réel sur les systèmes Unix et Unix-like, comme Linux. Elle affiche une liste dynamique des processus, triés par **utilisation du CPU** par défaut. Les informations affichées incluent **l'heure actuelle, le temps écoulé depuis le démarrage du système, le nombre d'utilisateurs connectés et les charges moyennes sur les dernières 1, 5 et 15 minutes.**

Dans la liste des processus, **top** affiche des détails tels que **l'ID du processus (PID), le nom de l'utilisateur, le pourcentage d'utilisation du CPU et de la mémoire (MEM)**.

L'outil offre également des **options de commande interactives** pour modifier son comportement en temps réel. Par exemple, vous pouvez naviguer dans la liste des processus, trier selon différents critères ou tuer des processus.

**Top** est essentiel **pour surveiller l'utilisation des ressources système, diagnostiquer les problèmes de performance et détecter les processus gourmands en ressources**. C'est un outil de base pour les administrateurs système et les utilisateurs avancés sur les systèmes Unix et Linux.



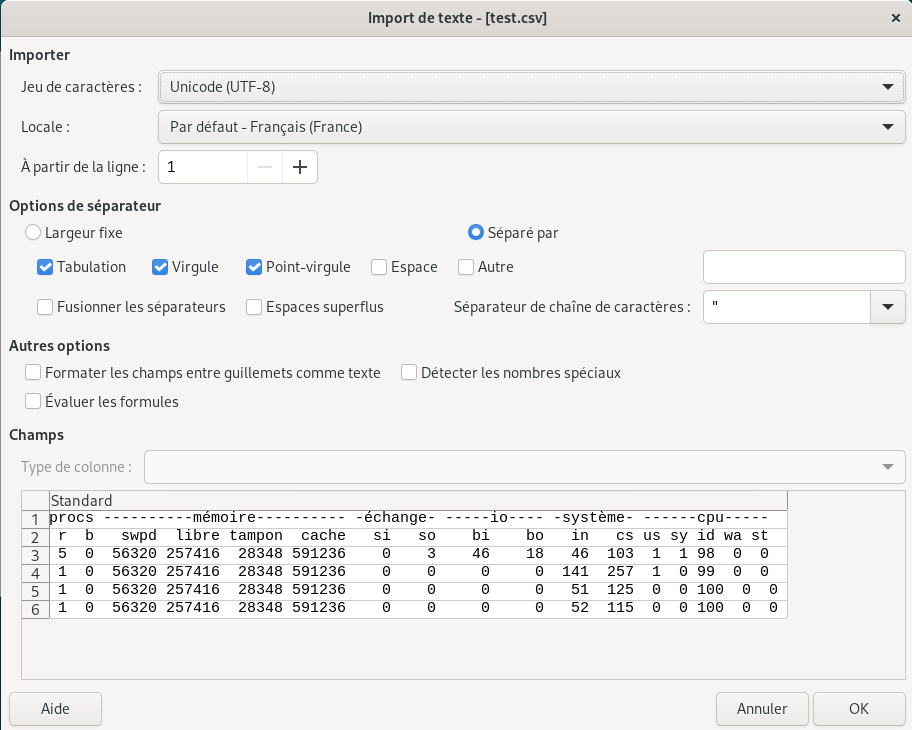
Enregistrer information dans un fichier csv   
Lorsque vous spécifiez **1** et **4** après la commande **vmstat**, cela signifie que **vmstat** va générer des statistiques à intervalles réguliers et pour un nombre spécifique de fois.

Plus précisément :

* Le **1** indique à **vmstat** d'effectuer une mesure à chaque seconde. Cela signifie que **vmstat** va générer une nouvelle ligne de statistiques toutes les secondes.
* Le **4** indique que **vmstat** va générer des statistiques pendant 4 cycles. Ainsi, **vmstat** s'exécutera pendant 4 secondes et générera une ligne de statistiques à chaque seconde.



Voici le fichier csv avec les informations de surveillance du système



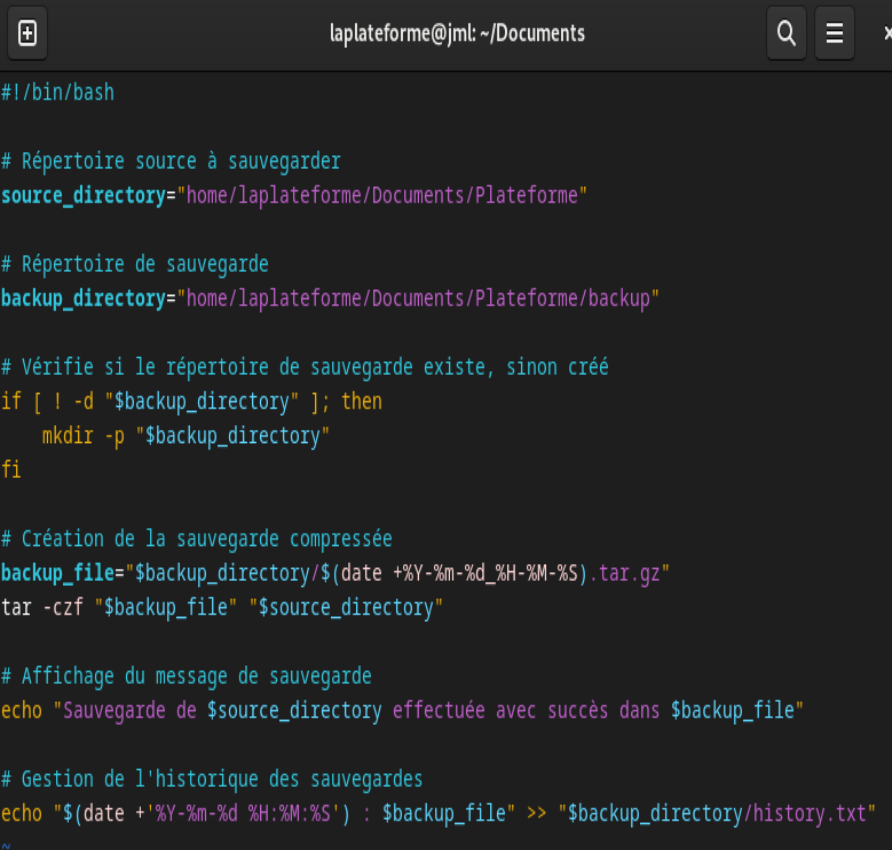
7- Scripting avancé

Développer un script Shell visant à automatiser la sauvegarde périodique du répertoire « Plateforme » créé précédemment. Assurez-vous d'intégrer une fonctionnalité de gestion de l'historique des sauvegardes, permettant ainsi de conserver un suivi chronologique des opérations effectuées.

Voici notre script pour automatiser la sauvegarde



Voici un deuxieme script possible :



Voici la commande pour accéder à notre fichier **Crontab**, bien mettre **sudo**





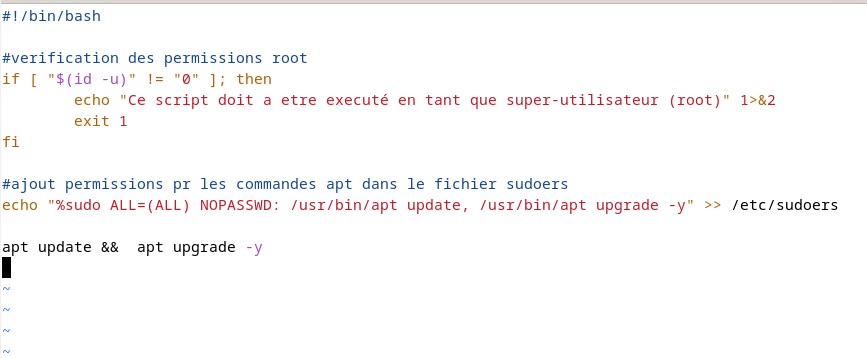
* Le premier champ (**\***) représente les minutes. Dans ce cas, **\*** signifie "toutes les minutes", ce qui signifie que le script sera exécuté chaque minute.
* Le deuxième champ (**\***) représente les heures. Comme il est également défini à **\***, cela signifie que le script sera exécuté à toute heure.
* Le troisième champ (**\***) représente les jours du mois. Encore une fois, étant donné qu'il est défini à **\***, cela signifie que le script sera exécuté tous les jours du mois.
* Le quatrième champ (**\***) représente les mois. Comme il est défini à **\***, cela signifie que le script sera exécuté tous les mois.
* Le cinquième champ (**\***) représente les jours de la semaine. Étant donné qu'il est défini à **\***, cela signifie que le script sera exécuté tous les jours de la semaine.

En résumé, cette ligne de cron planifie l'exécution du script **/home/manon/Documents/scriptSave.sh** toutes les minutes, tous les jours de l'année.

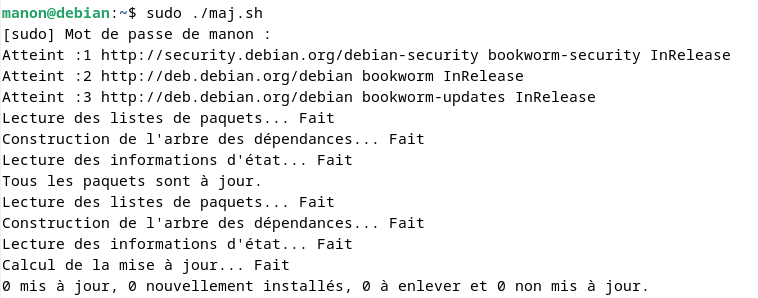
Commande pour **stopper Cron :**

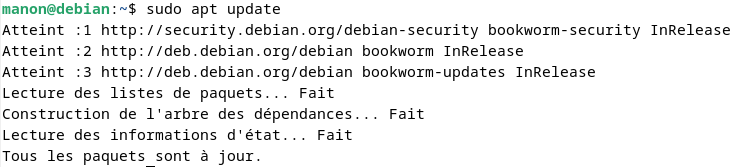


8- Automatisation des mises à jour logicielles

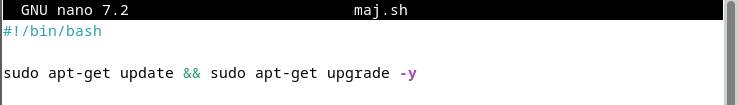




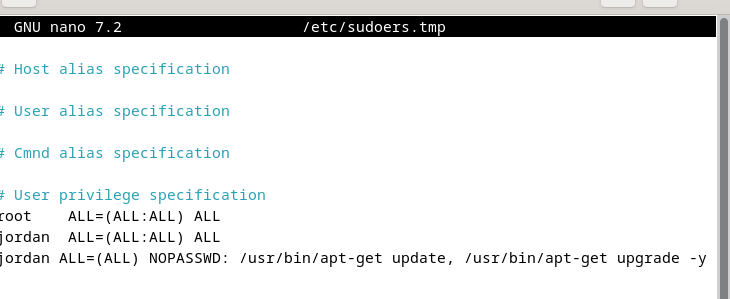




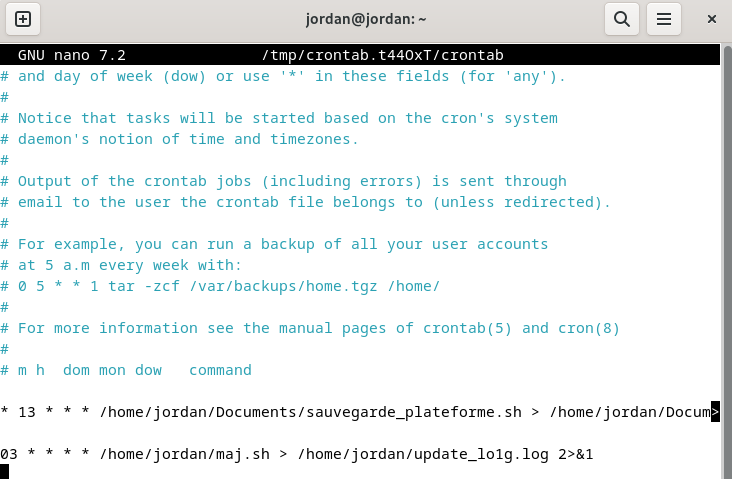
Option 2 :



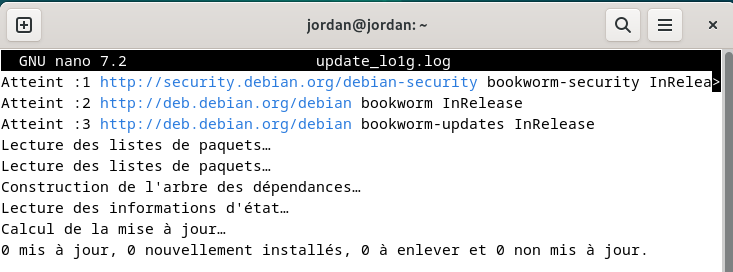
Droit sudoers pour autoriser les maj sans mdp



Fichier crontab



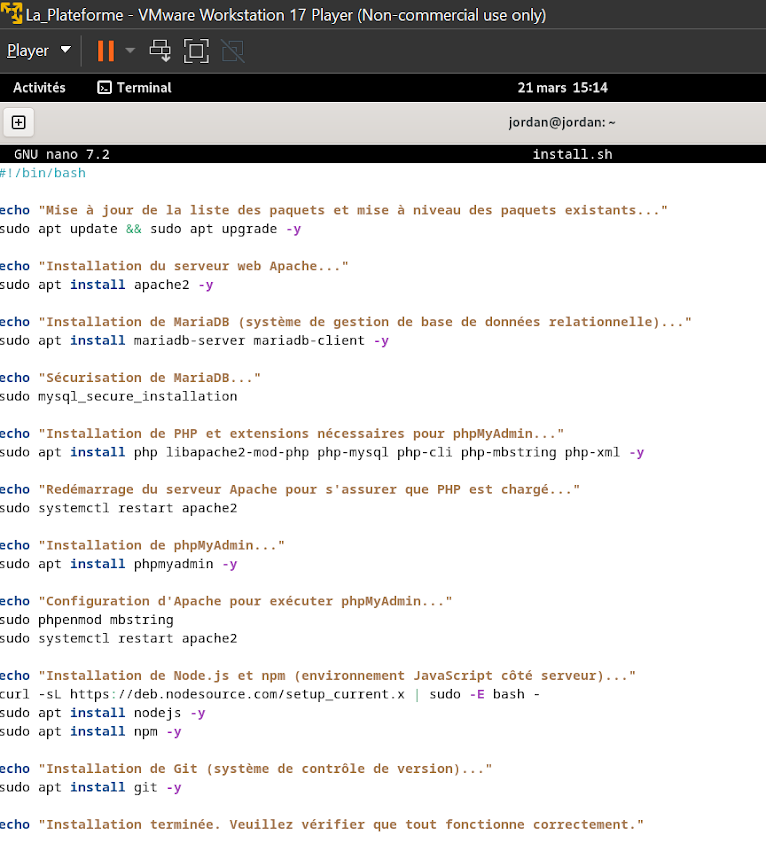
Log cron



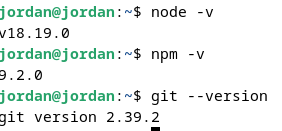
**9- Gestion des dépendances logiciels**

Voici le script ayant pour objectif de simplifier l’installation et la gestion des dépendances logicielles pour un projet web, tout assurant la compatibilité entre les différentes versions. Le script doit installer les éléments suivants :

* **Apache/Nginx** : Ce sont des serveurs web qui permettent de gérer les requêtes HTTP et de servir des pages web aux utilisateurs. Ils peuvent servir des fichiers statiques (comme les pages HTML et les images) et dynamiques (comme les pages PHP).
* **phpMyAdmin** : C'est une interface web pour gérer des bases de données MySQL et MariaDB. Elle permet d'effectuer des opérations sur les bases de données, comme la création de tables, l'exécution de requêtes SQL, l'importation et l'exportation de données, etc.
* **MariaDB** : C'est un système de gestion de base de données relationnelle qui permet de stocker, d'organiser et de récupérer des données de manière efficace.
* **Node.js** : C'est un environnement d'exécution JavaScript côté serveur qui permet d'exécuter du code JavaScript en dehors d'un navigateur web. Il est souvent utilisé pour créer des serveurs web, des outils de construction et des applications réseau.
* **npm** : C'est le gestionnaire de paquets pour Node.js. Il permet d'installer et de gérer des bibliothèques de code JavaScript qui peuvent être utilisées dans des projets Node.js.
* **Git** : C'est un système de contrôle de version décentralisé qui permet de suivre les modifications apportées aux fichiers d'un projet au fil du temps. Il est souvent utilisé pour la gestion de code source dans les projets de développement logiciel.

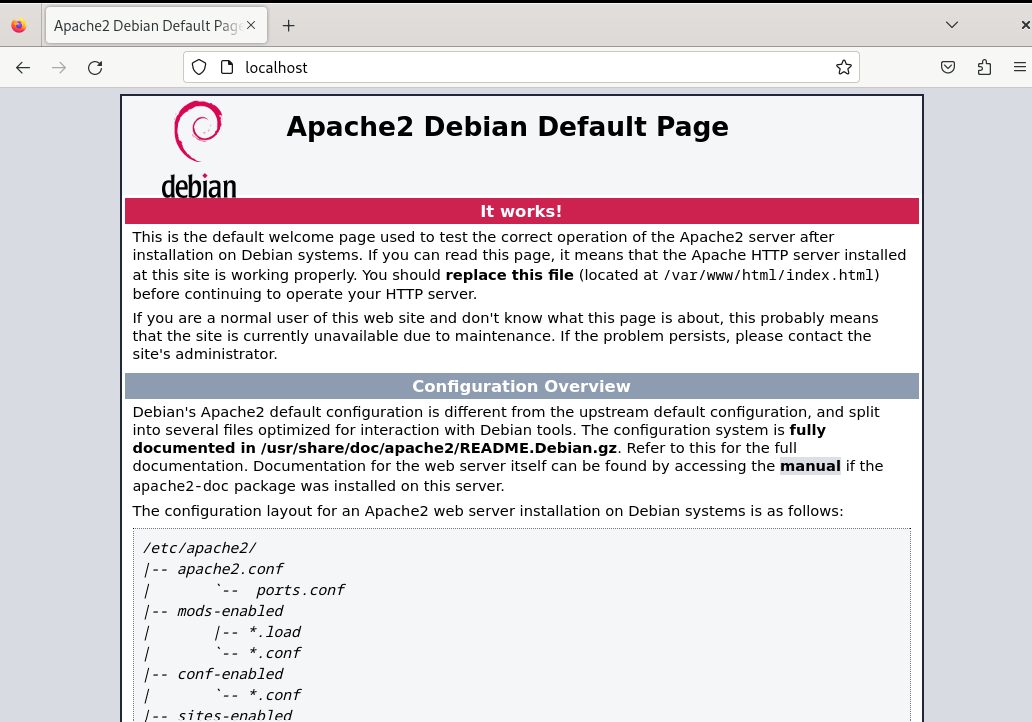
****

L'option **-y** est une option utilisée dans le terminal Linux pour **répondre automatiquement "oui"** à toutes les questions posées par certaines commandes. Cela permet d'automatiser les tâches et d'éviter d'avoir à confirmer manuellement chaque action. Par exemple, apt-get install -y <package> installe un paquet sans demander de confirmation. En résumé, -y est utilisé pour automatiser les réponses aux questions posées par les commandes et pour gagner du temps.

* ****
* node -v : affiche la version actuellement installée de Node.js sur votre système.
* npm -v : affiche la version actuellement installée de npm (Node Package Manager) sur votre système.
* git --version : affiche la version actuellement installée de Git sur votre système.

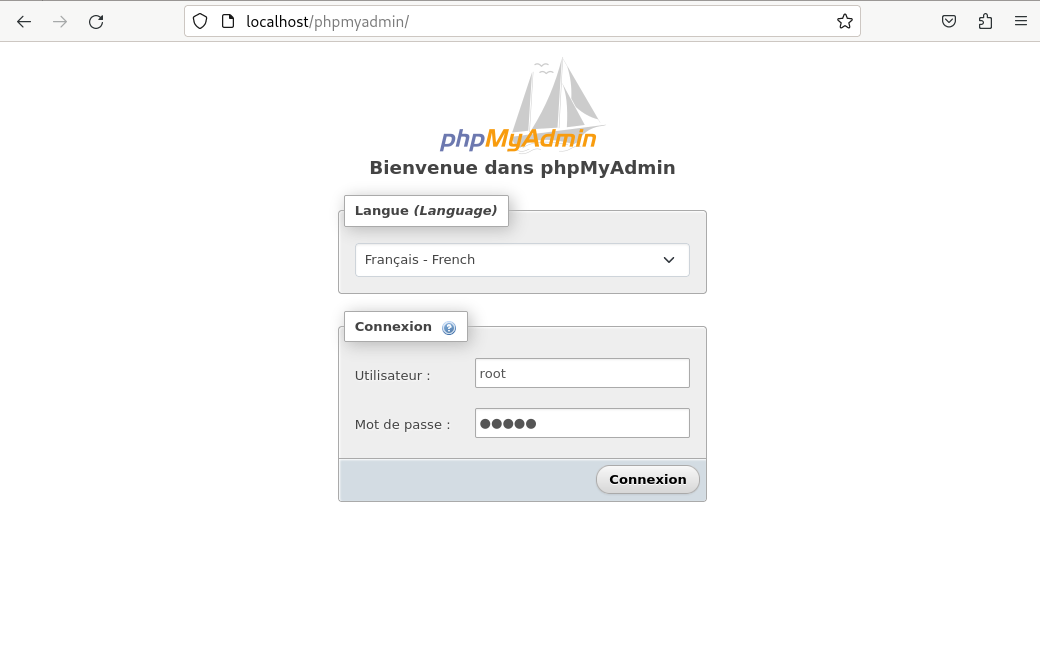
En résumé, ces commandes permettent de vérifier les versions respectives de Node.js, npm et Git installées sur votre système. Ces informations peuvent être utiles pour diagnostiquer des problèmes de compatibilité ou pour s'assurer que vous utilisez les versions les plus récentes et les plus stables de ces outils.

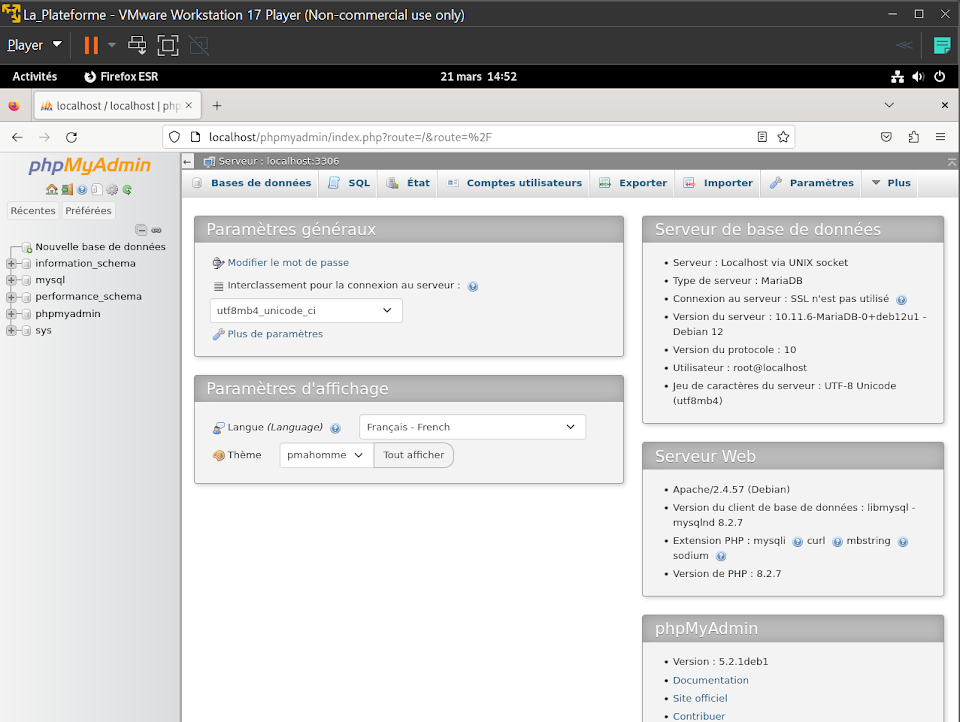
Maintenant nous pouvons aller sur **Firefox** et tapez [**http://localhost/**](http://localhost/)**,** pour voir si le serveur apache2 est bien installé



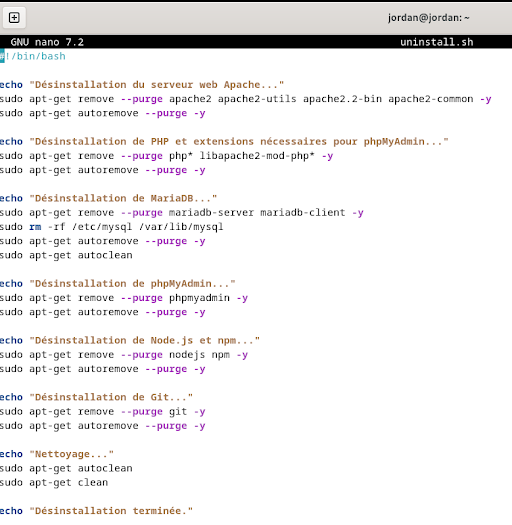
Pour voir phpMyadmin il suffit d’aller sur firefox est tapé [**http://localhost/phpmyadmin/**](http://localhost/phpmyadmin/)

On arrive sur la page de connexion rentrer l’utilisateur et le mot de passe definit



****

Nous avons également fait un Script de désinstallations des logiciels :

****

La commande **apt purge** est utilisée pour supprimer complètement un paquet et ses fichiers de configuration associés du système. Contrairement à **apt remove**, qui supprime uniquement les fichiers binaires du paquet, apt-get purge supprime également les fichiers de configuration. Cela garantit une désinstallation complète du paquet, idéale pour nettoyer le système des applications inutilisées.

**10- Sécuriser ses scripts**

Procédez à la sécurisation des différents scripts développés précédemment.

Pour sécuriser nos scripts, nous avons choisi de passer par **shc** qui est un compilateur de scripts shell qui permet de convertir un script shell en un binaire exécutable. Il prend en charge les scripts shell écrits dans les langages de script courants tels que Bash, sh, ksh, etc.

Lorsque vous compilez un script shell à l'aide de **shc**, le code source du script est converti en code machine et intégré dans un **programme C** (**gcc**). Le programme C est ensuite compilé en un binaire exécutable. Lorsque le binaire exécutable est exécuté, il extrait le code machine du script shell et l'exécute dans un interpréteur shell.

L'utilisation de **shc** peut offrir des avantages en termes de sécurité, car le code source du script shell est caché dans le binaire exécutable. Cela peut empêcher les utilisateurs non autorisés de lire ou de modifier le code source du script shell. De plus, le binaire exécutable peut être copié et exécuté sur d'autres systèmes sans avoir besoin de l'interpréteur shell ou des bibliothèques associées.

Cependant, il convient de noter que la compilation d'un script shell à l'aide de **shc** peut rendre le script plus difficile à déboguer et à maintenir, car le code source n'est plus disponible sous forme de texte lisible. De plus, le binaire exécutable peut être plus volumineux que le script shell d'origine.

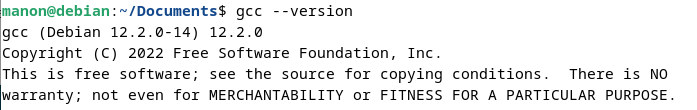
En résumé, **shc** est un compilateur de scripts shell qui permet de convertir un script shell en un binaire exécutable. Cela peut offrir des avantages en termes de sécurité, mais peut rendre le script plus difficile à déboguer et à maintenir.

Pour se faire il faut installer **shc** et vérifier si le programme C qui **gcc** est bien installé normalement oui

Commande pour installer **shc** :



Commande pour vérifier si le **programme C gcc** est bien installé



C’est quoi un programme C :

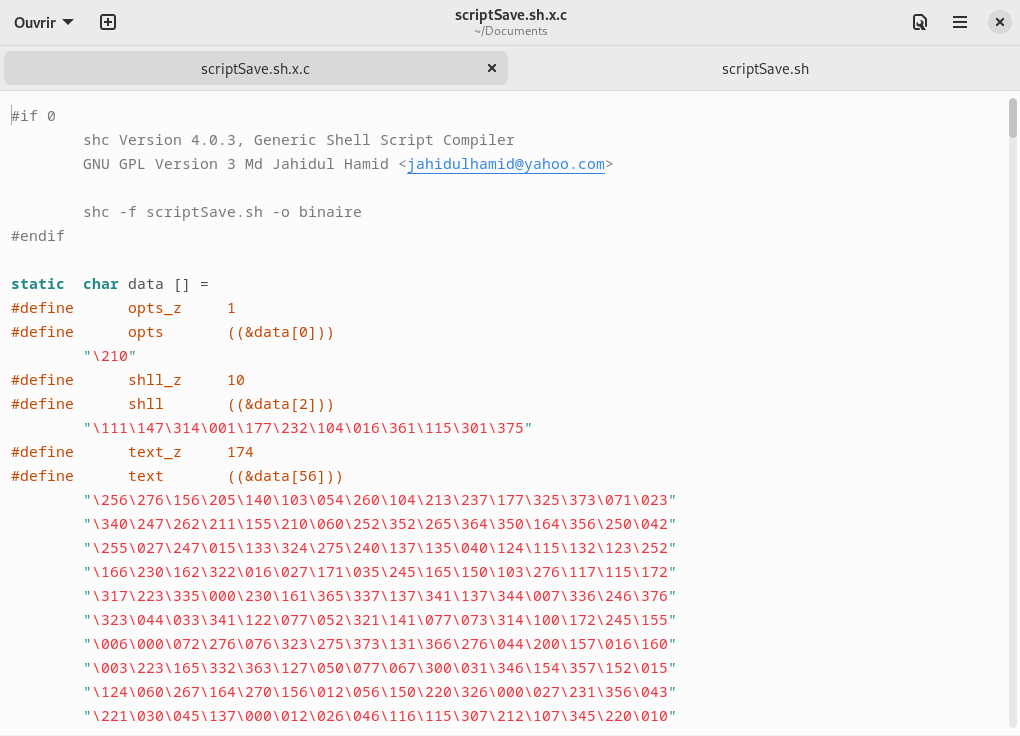
Le programme C est un programme écrit dans le langage de programmation C. Le langage C est un langage de programmation impératif et procédural qui a été développé dans les années 1970. Il est largement utilisé pour la programmation système, la programmation de jeux vidéo, la programmation embarquée, etc.

Le programme C est généralement compilé en code machine exécutable à l'aide d'un compilateur C tel que **gcc**.



La commande **compile un script shell scriptSave.sh en un binaire exécutable** à l'aide du compilateur de scripts shell shc. Le binaire peut être exécuté directement sur la ligne de commande et extrait le code machine du script shell pour l'exécuter dans un interpréteur shell.

Fichier script compilé en gcc :

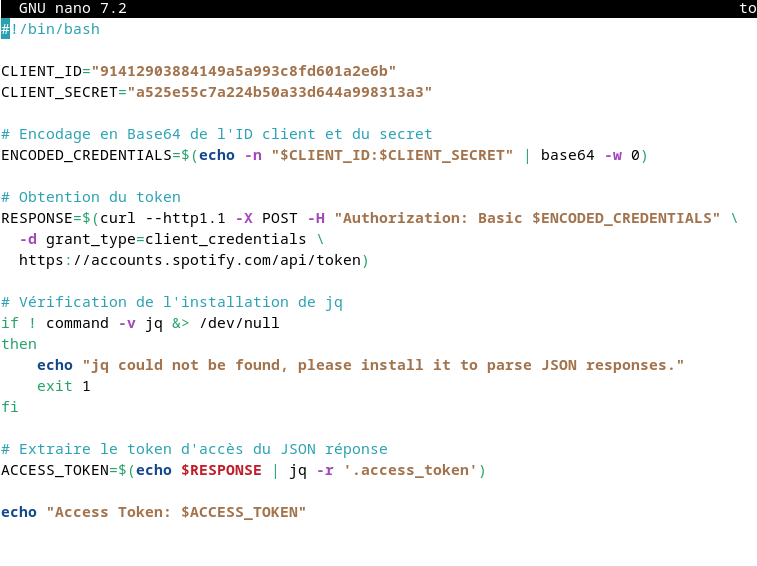


* Valider les entrées utilisateur pour éviter les injections de code et d'échapper correctement les caractères spéciaux dans les commandes système.
* Limiter les privilèges d'exécution, de lecture et d'écriture des scripts aux utilisateurs autorisés en utilisant des mécanismes de contrôle d'accès (chmod).
* Créer des groupes d'utilisateurs qui auraient le droit d'exécuter certains scripts sans pourtant les modifier ni même les lire.
* Effectuer régulièrement des analyses avec des logs (rsyslog, analyse des actions de cron etc), en s'assurant que les appels système sont sécurisés et surveillez les journaux pour détecter toute activité suspecte.
* Sécuriser l'accès aux logs avec les autorisations (chmod).
* Maintenir un système à jour avec les derniers updates pour prévenir les failles.

En suivant ces pratiques de sécurité, vous pouvez réduire les risques de vulnérabilités et d'attaques potentielles sur vos scripts shell et votre système.

**11- Utilisation d’API Web dans un script**

Nous avons créé un fichier pour récupérer le token de l’api que nous avons choisie

****

**C’est quoi un token** :

Un **token** est une chaîne de caractères qui représente une autorisation d'accès à une ressource protégée, telle qu'une API. Dans le cas de l'API Spotify, un token est nécessaire pour accéder aux données de l'utilisateur et effectuer des actions en son nom, telles que lire sa bibliothèque musicale ou contrôler la lecture.

Pour récupérer un token sur l'API Spotify, vous devez suivre les étapes suivantes :

1. Créer une application Spotify Developer et obtenir les identifiants de l'application (ID client et secret client).
2. Obtenir l'autorisation de l'utilisateur en redirigeant l'utilisateur vers la page de connexion Spotify et en demandant les autorisations nécessaires.
3. Échanger le code d'autorisation contre un token d'accès en envoyant une requête HTTPS POST à l'API Spotify avec les identifiants de l'application et le code d'autorisation.
4. Utiliser le token d'accès pour accéder aux ressources de l'API Spotify en envoyant des requêtes HTTPS avec le token dans l'en-tête d'autorisation.

Le token d'accès a une durée de vie limitée et doit être renouvelé périodiquement en utilisant le rafraîchissement de token.

**C’est quoi une API :**

Une **API (**Application Programming Interface) est un ensemble de protocoles, de définitions de données et de routines qui permettent à différents logiciels de communiquer entre eux et d'échanger des données.

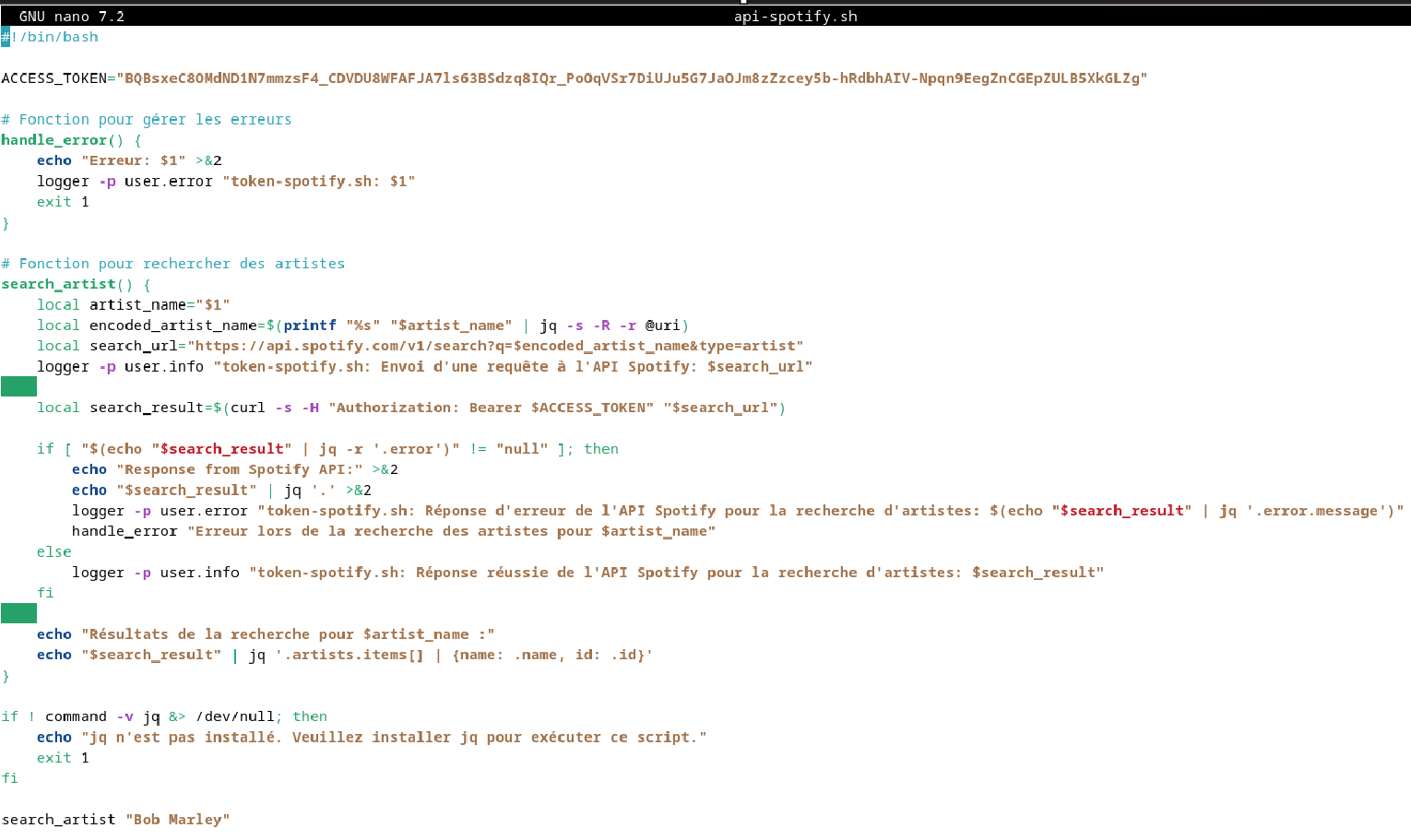
Une API définit les méthodes, les paramètres et les formats de données que les logiciels doivent utiliser pour communiquer entre eux. Les logiciels qui utilisent une API peuvent être des applications, des bibliothèques, des services web, etc.

Les API sont utilisées pour de nombreuses raisons, notamment :

1. Intégration de fonctionnalités externes : les API permettent aux développeurs d'intégrer des fonctionnalités externes à leur application, telles que la cartographie, la météo, la recherche, etc.
2. Échange de données : les API permettent aux applications d'échanger des données entre elles, telles que des données utilisateur, des données de transaction, etc.
3. Automatisation de tâches : les API permettent aux développeurs d'automatiser des tâches répétitives, telles que la création de rapports, la sauvegarde de données, etc.
4. Extension de fonctionnalités : les API permettent aux développeurs d'étendre les fonctionnalités de leur application en utilisant des services externes, tels que le traitement de paiement, la vérification d'identité, etc.

Les API peuvent être accessibles via différents protocoles, tels que HTTP, REST, SOAP, etc. Les développeurs peuvent utiliser des bibliothèques et des frameworks pour faciliter l'utilisation des API dans leur application.

Ce script nous permet d’exécuter une requête a l’api de Spotify nous permettant de faire des recherches sur différents artistes ou autres



Résultat recherche artiste bob marley



Nous avons du installer **rsyslog** qui est un **système de journalisation** avancé utilisé dans les systèmes basés sur Unix, notamment dans les distributions Linux. Il collecte, stocke et gère les journaux système, offrant des fonctionnalités étendues telles que le filtrage avancé, le routage des journaux, la journalisation sécurisée et une configuration flexible. **rsyslog** est largement utilisé pour surveiller et analyser les événements système, améliorant ainsi la sécurité, la surveillance et la gestion des systèmes informatiques

Résultat journalisation avec rsylog

