**Université Abdelmalek Essaadi**

**Ecole Nationale des Sciences Appliquées Al-Hoceima**

**04/01/2024**

Encadré par :

***Pr. Mohamed CHERRADI***

**Année Universitaire : 2023/2024**

***GHARIBI HANAN***

***EL KAMRI HAJAR***

***KHAYYI HANAE***

***KPATCHA Essohanam Josué***

***EL HADRATI OTHMAN***

Projet : Développement d'un système de fichiers et d'une distribution Linux personnalisée

Première Année Ingénierie des données

Réalisé par :

**Table des matières**

[Introduction PAGEREF \_Toc123945935 \h Error: Reference source not found](#_Toc123945935) ……………………………………………………...1

Notions Fondamentales :……………………………………..2

1. Les systèmes de fichiers :…...…………………2

***1.1. Définitions :……………………………………………………………..2***

***1.2. Catégories de fichiers :……………………………………………….3***

***1.3. Types de systèmes de fichiers :………………………………….....3***

2. Arch Linux :……………………...………………….5

***2.1. Définition :……………………………………………………………….5***

***2.2. Caractéristiques :……………………………………………………...5***

Mise en œuvre de notre distribution :………………………..6

***1. Conception du système de fichiers :…………………..6***

***1.1. La hiérarchie de l’ID1FS :…………………………………………...6***

***1.2. Architecture de l’ID1FS :……………………………………………7***

***1.3. Outils de développement :…………………………………………8***

***1.4. Implémentation de l’ID1FS :……………………………………….9***

* ***a) Commandes offertes par le système de fichiers***
* ***b) Journalisation***
* ***c) Métadonnées***
* ***d) Applications personnalisées de notre distribution***

***2. Conception de la distribution personnalisée :…….21***

***3. Intégration du système de fichiers :………………...28***

***4. Intégration des applications personnalisées :……30***

***5. Intégration du programme Translate :……………..31***

***6. Création de l’image ISO :……………….…………….31***

***C* *Conclusion :…………………………………………………..32***

***Remerciements : …………………………………………….33***

**Introduction :**

*Dans le vaste domaine de l'administration système, notre projet se fixe comme objectif majeur la création d'un système de fichiers innovant et son intégration harmonieuse dans une distribution Linux personnalisée. Cette aventure technique et conceptuelle se veut une exploration approfondie des aspects techniques et conceptuels liés à l'administration système.*

*Ce rapport détaille les étapes majeures de notre démarche, débutant par la conception minutieuse du système, suivie de son intégration dans la distribution Linux personnalisée, jusqu'à la création d'une image ISO et l'installation sur une machine virtuelle.*

1. **Notions Fondamentales:**

**1. Les systèmes de fichiers:**

**1.1 Définition:**

 Un système de fichiers est un système de classement, d’organisation sur un support de stockage qui structure et organise l’écriture, la recherche, la lecture, le stockage, la modification et la suppression de fichiers d’une manière spécifique. Il est important que les fichiers puissent être identifiés sans erreurs grâce à leur classification et que les utilisateurs puissent accéder à leurs fichiers le plus rapidement possible.

Les systèmes de fichiers définissent également des paramètres tels que les suivants :

• Conventions de dénomination des fichiers

• Attributs des fichiers

• Contrôle(s) d’accès

Les systèmes de fichiers sont également un composant opérationnel important, agissant comme une interface entre le système d’exploitation et tous les lecteurs connectés (internes et externes, par exemple via un port USB).

**2.2 Catégories des fichiers:**

On a plusieurs catégories des fichiers, parmi eux on cite :

* **Fichiers normaux:**

\* texte : courrier, sources des programmes, scripts, configuration ...

\* exécutables : programmes en code binaire …

* **Fichiers répertoires:**

Ce sont des fichiers conteneurs qui contiennent des références à d'autres fichiers. Véritable charpente de l'arborescence, ils permettent d'organiser les fichiers par catégories.

* **Fichiers spéciaux:**

Situés dans **/dev**, ce sont les points d'accès préparés par le système aux périphériques. Le montage va réaliser une correspondance de ces fichiers spéciaux vers leur répertoire "point de montage".

* **Fichiers liens symboliques:**

Ce sont des fichiers qui ne contiennent qu'une référence (un pointeur) à un autre fichier. Cela permet d'utiliser un même fichier sous plusieurs noms sans avoir à le dupliquer sur le disque.

**2.3 Types de systèmes de fichiers:**

Au fil des années, la diversification des systèmes de fichiers standard pour **Windows**, **MacOs**, **Linux**, **Unix** & **Co**. s'est accentuée, notamment en réponse aux progrès technologiques tels que le développement de systèmes sur mesure adaptés aux supports de mémoire flash de plus en plus répandus tels que les clés USB et les lecteurs SSD. Bien que chaque système de fichiers ait sa propre spécificité, tous reposent sur une structure arborescente, débutant par le répertoire racine et se ramifiant en dossiers, sous-dossiers et fichiers.

Parmi les nombreux systèmes de fichiers existants, seuls quelques-uns sont largement adoptés. Actuellement, les plus couramment utilisés comprennent **FAT16**, **FAT32**, **exFAT** et **NTFS** pour Windows, ainsi que **HFS+** et **APFS** pour **macOS**/**Mac OS X**. Du côté de Linux, le système **ext4**, successeur des **ext3** et **ext2**, est largement préféré.

Cette diversité reflète la nécessité d'adapter les systèmes de fichiers aux évolutions technologiques, en particulier face à la popularité croissante des supports de mémoire flash. Chaque système à ses avantages et ses limitations, et le choix dépend souvent des besoins spécifiques de l'utilisateur et de la compatibilité avec la plateforme utilisée.

Les systèmes de fichiers mentionnés sont brièvement présentés dans le tableau ci-dessous :

|  |  |  |  |
| --- | --- | --- | --- |
| *Systèmes de fichiers* | *Utilisation* | *Systèmes d’exploitation* | *caractéristiques* |
| *FAT32* | *Les supports de données mobiles* | *- Windows - Mac OS X/macOS - Linux (installer les pilotes si nécessaire)* | *- haute compatibilité - un large support materiel. - pas de fonctions de chiffrement et de compression. - La sécurité des données n'est pas particulièrement mise en avant. - idéal pour les petites partitions - taille maximale du fichier : 4 Go* |
| *ExFAT* | *Les supports de données mobiles.* | *- Windows - Mac OS X/macOS (Compatible à partir 10.6.4) - Linux (installer les pilotes si nécessaire).* | *- pas encore un standard commun. - pas de gestion des droits - pas de compression de données - idéal pour les petites mémoires flash à partir de 32 Go (clés USB, cartes SD) - tailles et partitions non limitées (selon l’état actuel de la technique) - taille maximale du fichier : 512 téraoctets* |
| *NTFS* | *Les disques durs internes et externes* | *- Windows - Mac OS X/macOS (support complet uniquement avec un outil supplémentaire) - Linux (après l'installation des pilotes)* | *- administration juridique - amélioration de la sécurité des données : protection contre la perte et la modification des données, possibilité de chiffrement des données. - ne convient pas aux petits disques et aux partitions de moins de 400 Mo (trop d’efforts de gestion). - taille maximale du fichier : 256 To* |

|  |  |  |  |
| --- | --- | --- | --- |
| *EXT4* | *Linux* | *- Linux - Windows (uniquement avec un outil supplémentaire) - Mac OS X/macOS (uniquement avec un outil supplémentaire)* | *Par rapport aux versions Ext précédentes : - amélioration des performances - amélioration de la sécurité des données - chiffrement intégré (à partir du noyau Linux 4.1) - gestion des droits possible. - taille maximale du fichier : 16 To* |

**2. ArchLinux:**

**1.1 Définition:**

 **Arch Linux** est une distribution de système d'exploitation Linux créée pour offrir un environnement minimaliste, personnalisable et axé sur la simplicité. Contrairement à d'autres distributions qui peuvent privilégier la facilité d'utilisation immédiate, **Arch Linux** adopte un modèle où l'utilisateur construit son système selon ses besoins spécifiques.

**2.2 Caractéristiques:**

**Rolling Release :** Arch Linux utilise un modèle de publication en continu (rolling release), ce qui signifie que les mises à jour sont publiées au fur et à mesure qu'elles sont disponibles, offrant ainsi un système constamment à jour.

**Simplicité et Minimalisme :** Arch Linux suit le principe du KISS (Keep It Simple, Stupid), favorisant la simplicité et la minimisation des composants préinstallés pour permettre à l'utilisateur de construire un système sur mesure.

**PacMan Package Manager :** Arch utilise le gestionnaire de paquets PacMan qui simplifie l'installation, la mise à jour et la suppression des logiciels.

**Arch User Repository (AUR) :** L'AUR permet aux utilisateurs de soumettre leurs propres scripts d'installation pour des logiciels tiers, élargissant considérablement la gamme de logiciels disponibles.

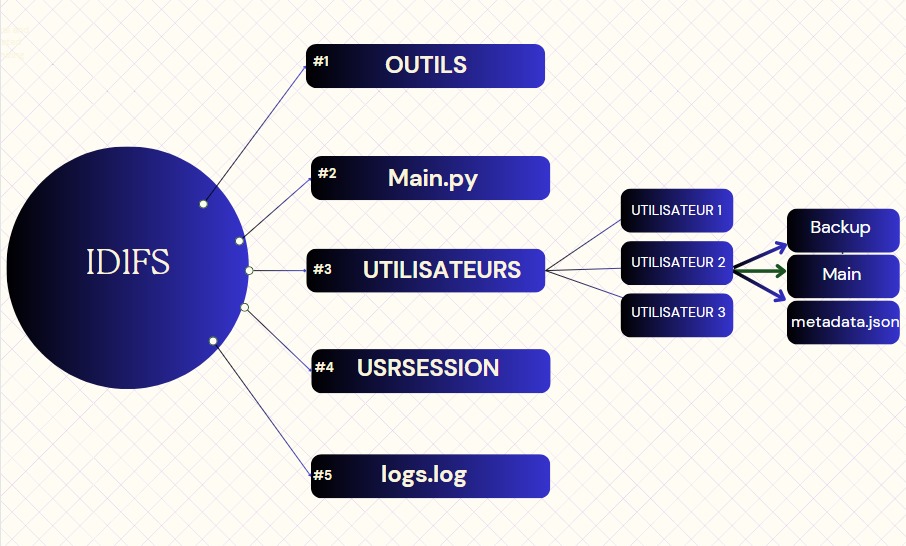
**Documentation Excellente** **:** Arch Linux offre une documentation complète (Arch Wiki) qui guide les utilisateurs à travers toutes les étapes de l'installation et de la configuration.

**Configuration Manuelle :** Arch Linux donne aux utilisateurs un contrôle complet sur la configuration de leur système, exigeant souvent une configuration manuelle des fichiers de configuration.

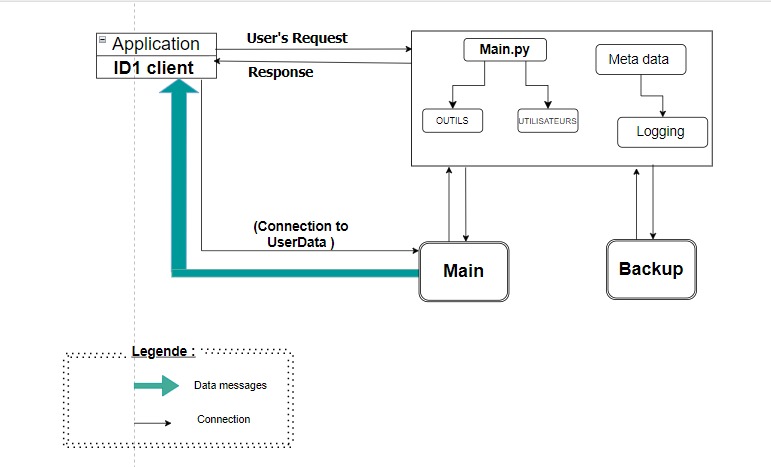
1. **Mise en œuvre de notre distribution :**

**1. Conception du système de fichiers :**

**1.1. La hiérarchie d’ID1FS :**



**1.2. L’architecture d’ID1FS :**



Dans cette Architecture on s'est inspirée de celle de <GOOGLE FILE SYSTEM>, Parmi les composants clés du système on distingue:

Le programme principal qui contient :

- **Main.py :** Un fichier system principal représentant le point d'entrer de programme qui gère les connexions, commandes de l'utilisateur et communique avec les autres composantes (fichiers) a savoir (OUTILS/UTILISATEUR...) afin de déterminer les résultats appropries.

- **Meta data :** ou on stocke les informations sur les fichiers de l'utilisateur.

- **Logging:** responsable de la journalisation.

+ Le programme principal à une relation directe avec les répertoires associés à chaque utilisateur tel que Main contenant les données de l'utilisateur et le Backup qui stocke les fichiers ou dossiers supprimés.

+ Les 2 répertoires (Main/Backup) ont une relation avec l'application client qui collecte les données nécessaires via des connexions les renvoie au Programme principal qui gère les résultats par la suite.

**1.3. Outils de développement :**

* **Langage de programmation utilisée :**

Python est un langage de programmation polyvalent, à la syntaxe claire et lisible, favorisant un développement rapide et efficace. Sa popularité réside dans sa capacité à répondre à divers besoins, de l'automatisation des tâches simples à des applications complexes en intelligence artificielle.

* **Bibliothèques utilisées:**

Parmi les bibliothèques de Python qu’nous avons utilisé :

* ***OS :*** La bibliothèque **os** en Python facilite l'interaction avec le système d'exploitation, offrant des fonctionnalités pour manipuler les fichiers, gérer les répertoires et exécuter des commandes système. Elle fournit une abstraction portable pour les opérations liées au système.
* ***Sys :*** La bibliothèque **sys** fournit des fonctionnalités permettant d'interagir avec l'interpréteur Python, notamment l'accès à des paramètres spécifiques du système et la manipulation du chemin de recherche des modules.
* ***json :*** La bibliothèque **json** en Python permet la sérialisation et la désérialisation de données au format JSON, facilitant l'échange de données structurées entre applications.
* ***Pandas :*** La bibliothèque **pandas** est une puissante bibliothèque d'analyse de données en Python, offrant des structures de données flexibles et des outils pour manipuler et analyser des tableaux de données.
* ***Argv :*** Le module **argv** fait partie de la bibliothèque **sys** et permet d'accéder aux arguments de la ligne de commande passés lors de l'exécution d'un script Python.
* ***Shutil :*** La bibliothèque **shutil** propose des utilitaires pour effectuer des opérations de haut niveau sur les fichiers et les répertoires, offrant des fonctionnalités telles que la copie, le déplacement et la suppression.
* ***Datetime*** : La bibliothèque **datetime** permet la manipulation d'objets de date et d'heure, offrant des fonctionnalités pour travailler avec des calendriers, des durées et des opérations de conversion.
* ***Logging :*** Le module **logging** fournit des fonctionnalités de journalisation flexibles, permettant aux développeurs de générer des messages de journal avec différents niveaux de gravité pour faciliter la gestion des informations dans une application.
* ***Subprocess*** : La bibliothèque **subprocess** offre des moyens d'exécuter des commandes système à partir de scripts Python, facilitant l'interaction avec le système d'exploitation.
* ***Getpass :*** Le module **getpass** permet de saisir en toute sécurité des mots de passe depuis la ligne de commande, masquant l'entrée utilisateur pour des opérations sensibles.
* ***Googletrans :*** La bibliothèque **googletrans** en Python offre une interface pour utiliser l'API de traduction de Google. Elle permet de traduire du texte entre différentes langues de manière simple et pratique.

**1.4. Implémentation ID1FS :**

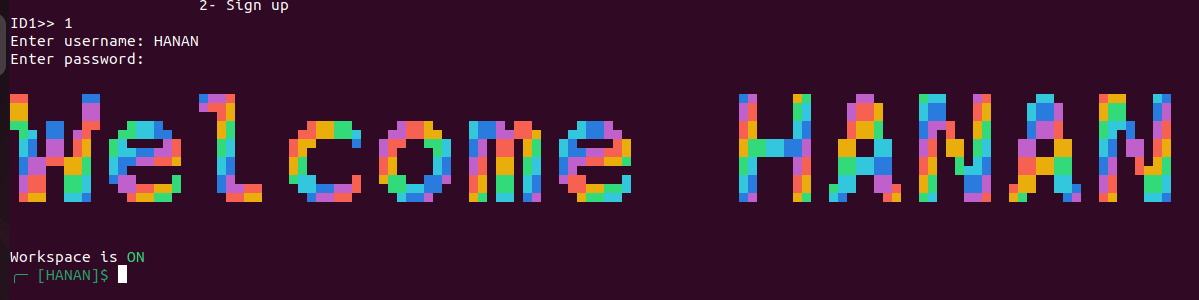
* **Démarrage du système de fichiers ID1FS :**



Après le démarrage du système de fichiers, une page de connexion s'affiche. Si vous possédez déjà un compte, vous tapez 1 ; sinon, vous tapez 2 pour créer un nouveau compte.



**1er cas :** Dans le cas d’une connexion réussie, la date de connexion est ajoutée à la ligne correspondant à cet utilisateur dans le fichier **"usrsession.txt".**



**2ème cas :** Lors de la création d'un compte, le nom d'utilisateur (username), le mot de passe subissant une fonction de hachage, ainsi que la date de connexion sont enregistrés dans un fichier texte, **"usrsession",** situé dans le dossier **"USRSESSION".** Simultanément, un répertoire personnel portant le nom d'utilisateur est créé dans le répertoire **"UTILISATEURS".**



Ce répertoire contient un dossier principal **"Main"** destiné à accueillir tous les fichiers ou dossiers créés par l'utilisateur, ainsi qu'un dossier de sauvegarde **"Backup"** agissant comme corbeille pour les fichiers supprimés, facilitant leur récupération.



**a.)Les commandes offertes par l’ID1FS:**

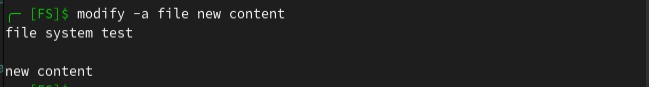
Ce sont des instructions qui exécutent des fonctions stockées dans des fichiers scripts .py, ces derniers sont rassemblés dans le dossier **OUTILS** en plus d’autres fichiers auxiliaires. Dans cette section, on explique chaque commande et ses fonctionnalités.

* **La commande modify :** La commande «**modify**» est principalement dédiée à l'écriture dans un fichier, avec une syntaxe simple : il suffit de saisir **modify**, suivi du nom du fichier à éditer et de l'option d'édition. À noter que si le fichier n'existe pas, il sera créé avant d'être modifié. Le nouveau contenu du fichier sera ensuite affiché à l'écran.

**Syntaxe : modify <nom du fichier> <option> <content>**

**Options associées :**

* **'-a' :** ajoute un nouveau contenu à un fichier existant**.**
* **'-w' :** écrase le fichier s'il existe, ou crée un nouveau fichier et y écrit le contenu.



C:\Users\hp\AppData\Local\Microsoft\Windows\INetCache\Content.Word\WhatsApp Image 2023-12-23 à 01.04.47_3536055a.jpg

* **La commande show :** La commande «**show**» est spécialement conçue pour afficher le contenu d'un fichier. Son utilisation est simple : il suffit de saisir **show** suivi du nom du fichier dont on souhaite afficher le contenu. Si le fichier existe, son contenu sera affiché à l'écran.

**Syntaxe : show <nom du fichier>**

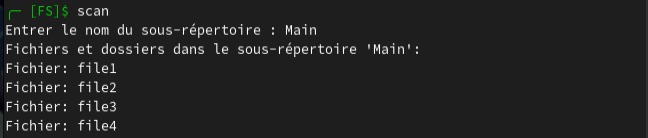
C:\Users\hp\AppData\Local\Microsoft\Windows\INetCache\Content.Word\IMG-20231223-WA0008.jpg

* **La commande rename :** La commande «**rename**» permet de renommer un fichier. La syntaxe est simple : il suffit de taper **rename**, suivi du nom actuel du fichier, puis du nouveau nom souhaité. Si le fichier ou le répertoire existe, il sera renommé conformément à la nouvelle désignation.

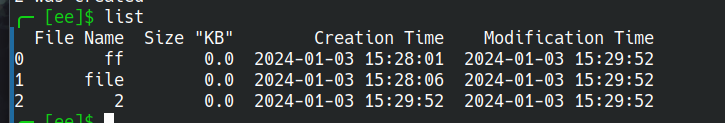
**Syntaxe : rename <nom du fichier à renommer> <nouveau nom>**

C:\Users\hp\AppData\Local\Microsoft\Windows\INetCache\Content.Word\IMG-20231223-WA0012.jpg

* **La commande scan :** La commande «**scan**» est utilisée pour explorer le contenu d'un sous-répertoire spécifié par l'utilisateur. La syntaxe consiste à entrer **scan**, suivi du chemin du sous-répertoire à explorer. Cette commande affiche ensuite une liste du contenu distinguant clairement s'il s'agit d'un fichier ou d'un dossier.

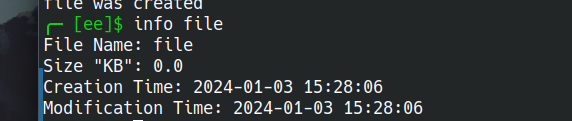


* **La commande list :** La commande «**list**» présente d’une façon détaillée des fichiers de l'utilisateur, incluant leurs métadonnées en lisant le fichier **Metadata.json**. La syntaxe est la suivante : **list**.



* **La commande info :** La commande «**info**» est conçue pour fournir des informations détaillées sur un fichier spécifié par l'utilisateur. Il suffit de taper **info**, suivi du nom d’un fichier.

**Syntaxe : info <nom du fichier >**



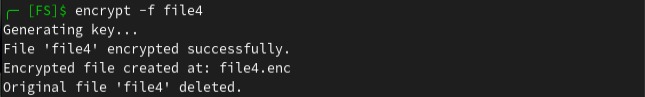
* **La commande show :** La commande «**show**» est destinée à l'affichage du contenu d'un fichier spécifié. Pour l'utiliser, entrez la commande suivante dans la ligne de commande **show** suivi du nom du fichier. Cette commande affiche de manière lisible le contenu du fichier spécifié directement dans la console.

**Syntaxe : show <nom du fichier >**

C:\Users\hp\AppData\Local\Microsoft\Windows\INetCache\Content.Word\show.jpg

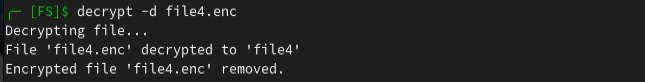
* **La commande encrypt :** La commande «**encrypt**» est dédiée à la sécurisation des fichiers via le chiffrement. Pour l'utiliser, il faut saisir **encrypt -f** suivi du nom du fichier en ligne de commande. Cette commande génère une clé de chiffrement est stockée dans un fichier appelé **'encryption\_key.key’**, puis chiffre le contenu du fichier spécifié. Le fichier original est supprimé, et une version chiffrée portant l'extension ‘.enc' est créée dans le même répertoire.

**Syntaxe : encrypt -f <nom du fichier >**



* **La commande decrypt :** La commande «**decrypt**» est dédiée à la récupération sécurisée des fichiers chiffrés. Pour l'utiliser, entrez la commande suivante **decrypt -d** suivi du nom du fichier chiffré. Cette commande charge automatiquement la clé de déchiffrement, puis décode le contenu du fichier chiffré spécifié. Le fichier déchiffré est enregistré avec le même nom que le fichier d'origine, sans l'extension '.enc', et le fichier chiffré est supprimé.

**Syntaxe : decrypt -d <nom du fichier >**



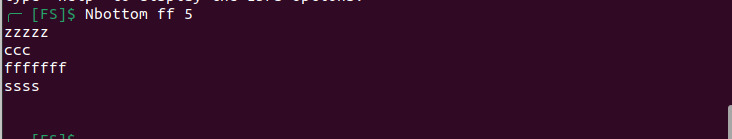
* **La commande Ntop :** La commande «**Ntop**» est conçue pour afficher les premières lignes d'un fichier spécifié. Pour utiliser cette commande, il faut saisir **Ntop** suivi du nom du fichier et du nombre de lignes que vous souhaitez afficher dans la ligne de commande.

**Syntaxe : Ntop <nom du fichier> <nombre de lignes>**



* **La commande Nbottom :** La commande «**Nbottom**» est conçue pour afficher les dernières lignes d'un fichier spécifié. Pour utiliser cette commande, il faut saisir **Nbottom** suivi du nom du fichier et du nombre de lignes que vous souhaitez afficher dans la ligne de commande.

**Syntaxe : Nbottom <nom du fichier> <nombre de lignes>**



* **La commande make :** La commande «**make**» est dédiée à la création de fichiers. Pour l’utiliser, il vous suffit de saisir **make**, suivi de l'option désirée et du nom du fichier ou des fichiers à créer simultanément.

**Options disponibles :**

-x : Crée un nouveau fichier uniquement s'il n'existe pas déjà de fichier portant ce nom.

-w : Écrase tout fichier existant trouvé avec le même nom de fichier.

C:\Users\hp\AppData\Local\Microsoft\Windows\INetCache\Content.Word\IMG-20231223-WA0005.jpgIMG-20231223-WA0016**Syntaxe make : <option> <nom du fichier1> <nom du fichier2>**

* **La commande del :** La commande «**del**» est destinée à la suppression de fichiers. Pour l'utiliser, saisissez **del**, suivi du nom du fichier ou des fichiers que vous souhaitez supprimer simultanément. Les fichiers sont ainsi déplacés vers un folder appelé **Backup** pour etre récupéré par la suite.

C:\Users\hp\AppData\Local\Microsoft\Windows\INetCache\Content.Word\IMG-20231223-WA0014.jpg**Syntaxe : del <nom du fichier>**

* **La commande reinstate :** La commande «**reinstate**» est destinée à restaurer des fichiers précédemment supprimés. Pour l'utiliser, saisissez **reinstate** et choisir le fichier à restaurer. Cette commande rétablit les fichiers supprimés dans leur état précédent, les rendant à nouveau accessibles et utilisables.



* **La commande setdir :** La commande «**setdir**» est conçue pour créer des répertoires. Pour l'utiliser, saisissez **setdir** suivi du nom du répertoire que vous souhaitez créer pour le créer dans le dossier **Main** ou son chemin.

**Syntaxe : setdir <nom ou chemin du dossier>**

IMG-20231223-WA0015

* **La commande remove :** La commande «**remove**» est conçue pour supprimer des répertoires. Pour l'utiliser, saisissez **remove** suivi du nom ou le chemin du répertoire que vous souhaitez supprimer. Il est ainsi déplacé vers le dossier **Backup**.

**Syntaxe : remove <nom ou chemin du dossier>**

C:\Users\hp\AppData\Local\Microsoft\Windows\INetCache\Content.Word\IMG-20231223-WA0008.jpg

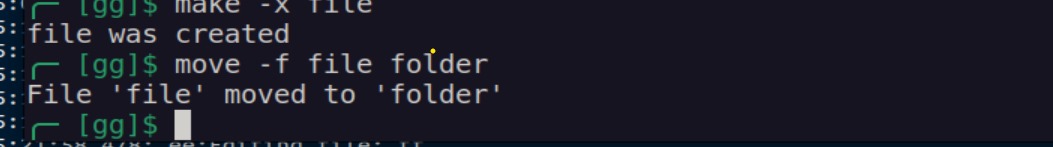
* **La commande move :** La commande «**move**» est utilisée pour déplacer des fichiers et des répertoires. Pour l’utiliser, il suffit de saisir **move**, suivi de l'option désirée et du chemin source du fichier ou répertoire à déplacer et le chemin de la destination.

**Options disponibles :**

-f : Pour déplacer des fichiers.

-d : Pour déplacer des répertoires.

**Syntaxe : move <option> <source> <destination>**



* **La commande upload :** La commande «**upload**» est utilisée pour le transfert d'un fichier depuis la machine vers le système de fichiers ID1FS. Pour l’utiliser, il faut saisir **upload** suivi par le chemin du fichier à recevoir.

**Syntaxe : upload <chemin du fichier Dans la distribution>**

C:\Users\hp\AppData\Local\Microsoft\Windows\INetCache\Content.Word\upload.jpg

* **La commande download :** La commande «**download**» est utilisée pour le transfert d'un fichier depuis le système de fichiers ID1FS vers la machine. Pour l’utiliser, il faut saisir **download** suivi par le fichier à transférer puis le chemin de la destination.

**Syntaxe : download <nom ou chemin du fichier> <destination de la distribution >**

WhatsApp Image 2023-12-23 à 01

* **La commande userdel :** La commande «**userdel**» est utilisée pour supprimer l’utilisateur et son répertoire**.** Il suffit de saisir **userdel** suivi du nom de l’utilisateur à supprimer.

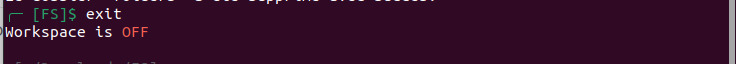
**Syntaxe : userdel <le nom de l’utilisateur à supprimer>**

C:\Users\hp\AppData\Local\Microsoft\Windows\INetCache\Content.Word\IMG-20231223-WA0006.jpg

* **La commande clear :** La commande «**clear**» est utilisée pour effacer ce qui est affiché dans la console**.** Il suffit de saisir **clear** suivi de ‘**all’** pour tout effacer ou du nombre de lignes à effacer.

**Syntaxe : clear < ’all’ ou le nombre de lignes>**

* **La commande exit :** La commande «**exit**» est utilisée pour sortir du système de fichiers**.** Il suffit de saisir **exit**.



* **La commande ToDoList :** La commande «**ToDoList**» offre une interface en ligne de commande (CLI) pour la gestion de listes de tâches. Vous pouvez ajouter de nouvelles tâches, les supprimer, les mettre à jour, les marquer comme terminées, et afficher la liste actuelle à tout moment. Le statut des tâches (complété ou non) est indiqué par des emojis (✅ ou ❌).

Un fichier **todos.db** est ainsi créé dans le répertoire USRSESSION pour stocker les données collectées par cette commande sous formes de tables.

* + - * **Ajout des taches :**

**Syntaxe : ToDoList add <Description><Catégorie>**

* + - * **Supression des taches :**

**Syntaxe : ToDoList delete <Position>**

* + - * **Mise à jour des taches :**

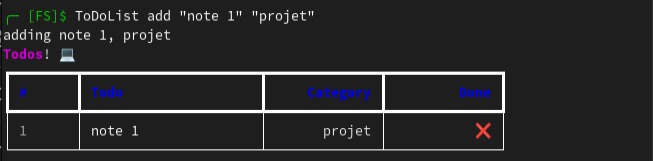
**Syntaxe : ToDoList update <Position><Description> <Catégorie>**

* + - * **Mise à jour du Statut des taches :**

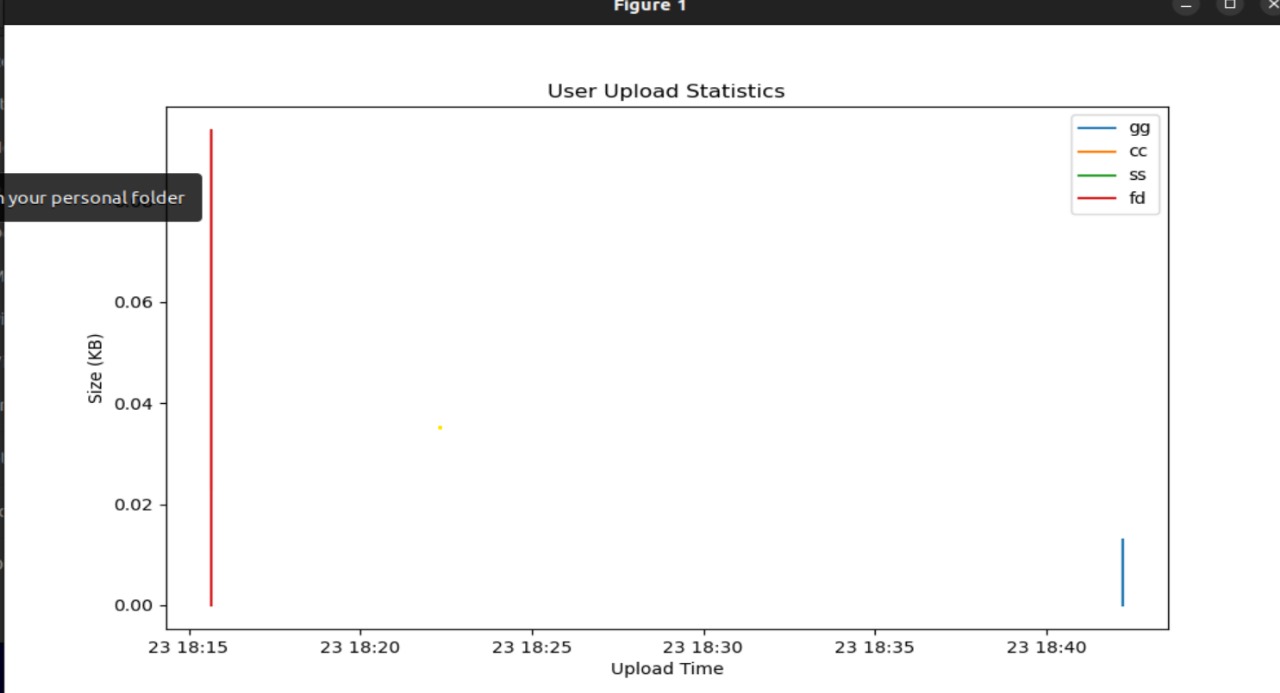
**Syntaxe : ToDoList complete <Position>**

* + - * **Affichage de la liste des taches :**

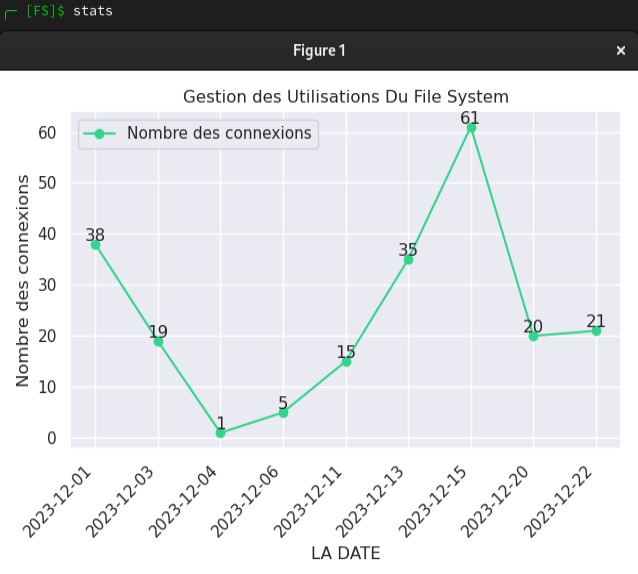
**Syntaxe : ToDoList show**



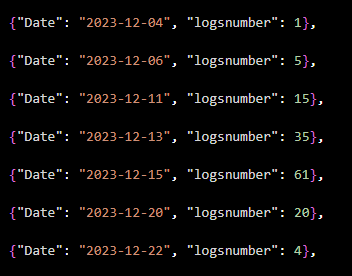
* **La commande status :** La commande «**status**» est utilisée pour avoir une idée générale sur le nombre de fichiers stockés par utilisateurs**.** Il suffit de saisir **status** pour afficher la visualisation graphique.



* **La commande stats :** La commande «**stats**» permet de visualiser le nombre de connexions par utilisateurs**.** Il suffit de saisir **stats** pour afficher le graphe



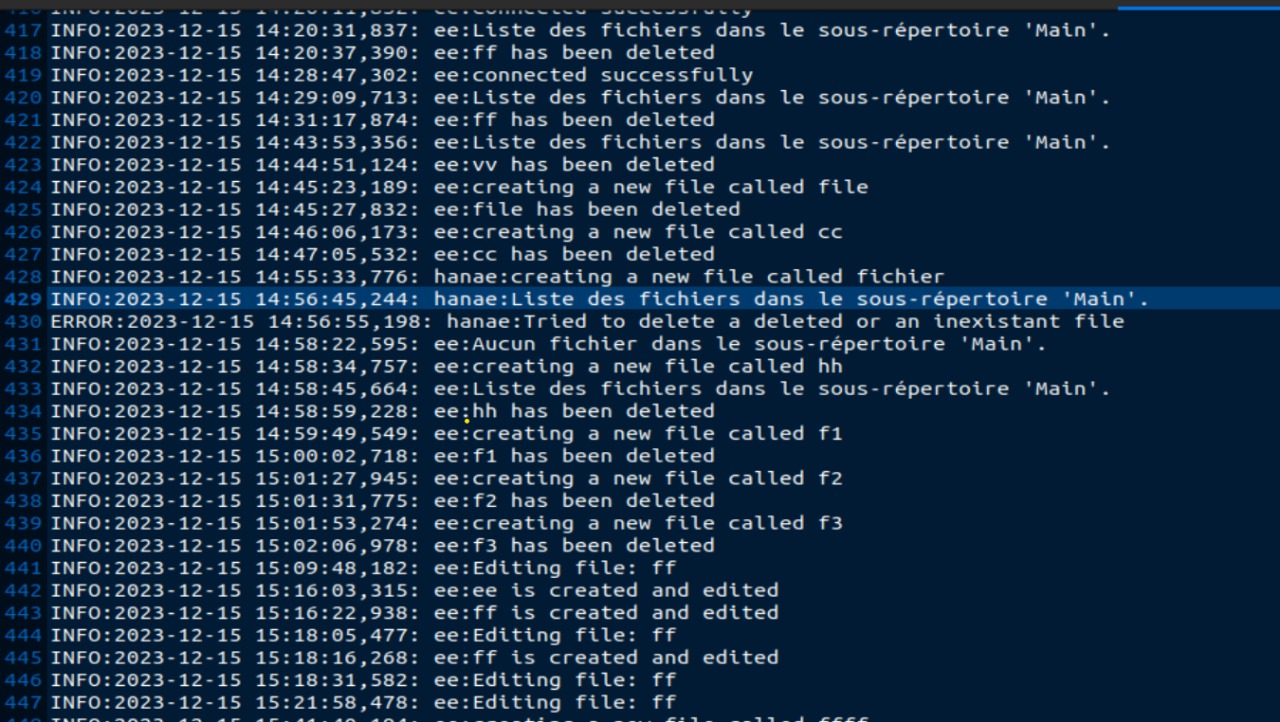
* **Ces deux commandes utilisent les données stockées dans un fichier nommé loginsnum.json situé dans le dossier OUTILS pour représenter ces graphes.**



**b.La journalisation :**

La journalisation ou **Logging** revêt une importance cruciale dans la gestion des systèmes de fichiers. En enregistrant chaque activité et erreur survenue, elle facilite le dépannage en permettant une identification rapide des problèmes. Ces journaux jouent un rôle clé en matière de sécurité, en détectant les activités suspectes et renforçant la résilience contre les intrusions.

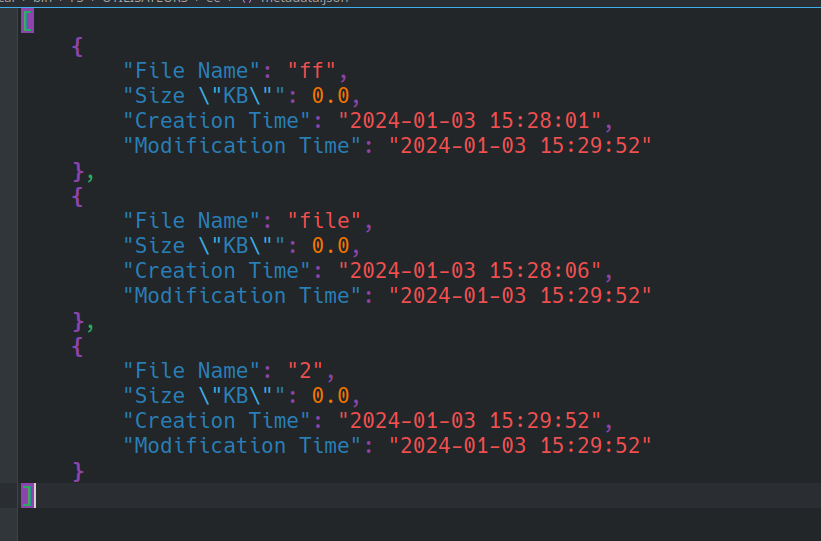
Voici un aperçu sur le fichier **logs.log** qui est conçue à enregistrer l’historique des activités survenus dans **ID1FS** de manière détaillée en listant la date et l’utilisateur correspondants ainsi que la nature de cette activité (INFO, ERROR).



**c.Les Métadonnées:**

Les métadonnées désignent des informations qui décrivent d'autres données. Elles fournissent des détails sur les caractéristiques, les propriétés des données qu'elles décrivent telles que le nom du fichier, sa taille, la date de sa création, et la date de sa dernière modification.

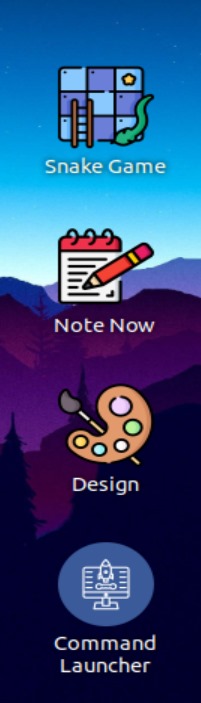
Dans le cadre de notre système de fichiers, on des fichiers .**json** appelés **Metadata.json**, dédiés à stocker les métadonnées des fichiers créés par chaque utilisateur de manière séparée. Chacun de ses fichiers est situé dans le répertoire personnel de l’utilisateur concerné.



**d.Applications personnalisées de notre distribution :**

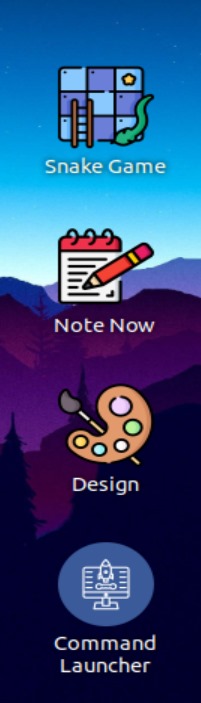
**Command Launcher**

L'application commande qui est liée au script python **" commande.py "** développée offre une interface conviviale permettant à l'utilisateur d'exécuter des commandes Linux de base via une interface graphique réalisée avec **Tkinter** en Python3. Les commandes sont exécutées à l'aide de la bibliothèque **subprocess**, offrant un moyen interactif et intuitif pour les utilisateurs. Chaque commande est associée à un bouton, déclenchant l'exécution correspondante avec des résultats affichés dans des boîtes de dialogue. L'utilisation de **Tkinter** garantit une disposition claire, tandis que **subprocess** facilite le lancement et la gestion des processus système. En somme, l'application combine efficacité et convivialité pour une expérience utilisateur optimale.



**Design**

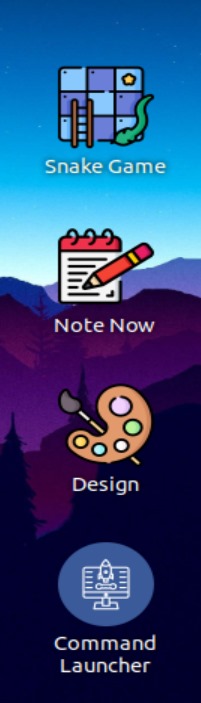
Le script Python **"Design.py"** présent constitue une application de dessin développée avec **Tkinter**. L'interface permet à l'utilisateur de dessiner sur un canevas à l'aide de la souris. Les fonctionnalités principales comprennent un bouton "Clear" pour effacer le dessin et une personnalisation visuelle avec une police spécifique et des couleurs. L'interaction utilisateur est intuitive, avec la souris utilisée pour démarrer et suivre le dessin. En conclusion, l'application offre une expérience de dessin légère et conviviale, illustrant l'utilisation pratique de **Tkinter** pour créer des interfaces graphiques interactives en Python.



**Note Now**

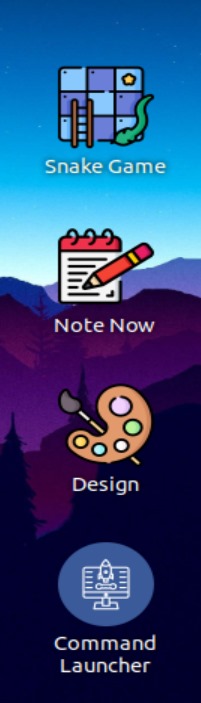
**« VIP.py"** présente une application de prise de notes développée avec **Tkinter**. L'interface

Le script Python **"note** permet à l'utilisateur de créer, sauvegarder et ouvrir des notes textuelles. Les fonctionnalités comprennent une zone de texte extensible, des boutons distincts pour sauvegarder et ouvrir des notes, et une esthétique visuelle soignée. L'utilisation de **Tkinter** simplifie la création de l'interface, offrant une solution conviviale pour la gestion de notes textuelles.



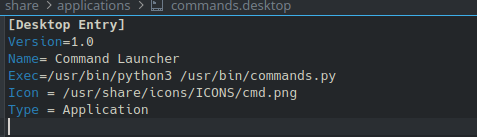
**Snake Game**

Le script Python **"relax.py"** utilise la bibliothèque **"Pygame"** pour implémenter le jeu classique du serpent. Les fonctionnalités clés comprennent le contrôle du serpent par le joueur, ainsi que l'affichage en temps réel du score. **"Pygame"** est utilisé pour la création de l'interface graphique, la gestion des événements clavier. Le script propose une expérience de jeu simple et divertissante tout en démontrant l'utilisation efficace de Pygame pour le développement de jeux en Python.



Pour L'optimisation des applications comme **"Launcher command"** dans les environnements de bureau du notre distribution on a créé un script de format **".desktop"**. Cette opération facilite l'accès à l'application en définissant le nom de chaque application, et en associant un script Python exécutable via la commande **"/usr/bin/python3 /usr/bin/nom-script.py".** L'icône soigneusement sélectionnée à **/usr/share/icons/ICONS/icon-d'application.png"** reflète la créativité de l'application. Cette conception permet aux utilisateurs d'accéder rapidement à l'application depuis leur environnement de bureau, améliorant ainsi l'expérience utilisateur et l'efficacité d'utilisation au quotidien.

L’exemple ci-dessous représente le cas pour l'application **COMMAND LAUNCHER** :



* Le dossier qui concerne les Applications contient divers fichiers et dossiers pour le fonctionnement et la personnalisation des applications sur le bureau. Voici une description détaillée de ces fichiers et dossiers :
* **Fichiers .py :** Ce sont des scripts pour les applications, qui permettent d'exécuter des tâches spécifiques à l'intérieur des applications. Par exemple **"noteVIP.py".**
* **Fichiers .desktop** : Ces fichiers sont utilisés pour transformer chaque fichier.py en une application dans le bureau. Par exemple, le fichier **"notepad.desktop"** pourrait être utilisé pour transformer le script **"noteVIP.py"** en une application **"Note Now"** dans le bureau.
* **Dossier ICONS :** Ce dossier contient les images utilisées dans les **fichiers.desktop** pour représenter les applications.
* **Dossier FONTS :** Ce dossier contient les polices utilisées dans les applications. Par exemple, le fichier **"Arial.ttf"** pourrait être une police utilisée dans les applications de ce dossier.

**2.Conception de la distribution personnalisée :**

**Objectifs de la distribution :**

Notre Distribution basée sur la filiale Arch linux a pour but principale l’intégration de notre système de fichiers dans un environnement linux propre à lui où on peut exploiter dans leurs plein puissances les fonctionnalités réalisées par le système de fichiers .

**Prérequis :**

* **Configuration du système Pour l’installation d’ Arch linux:**

Arch Linux devrait fonctionner sur n'importe quelle machine compatible x86\_64 avec un minimum de 512 MiB de RAM, bien que plus de mémoire soit nécessaire pour démarrer le système live lors de l'installation. Une installation de base devrait prendre moins de 2 GiB d'espace disque. Comme le processus d'installation doit récupérer des paquets depuis un référentiel distant, une connexion Internet fonctionnelle est requise. Ensuite aller sur le site [https://archlinux.org/download/](https://wiki.archlinux.org/title/Installation_guide) pour télécharger l’image iso.

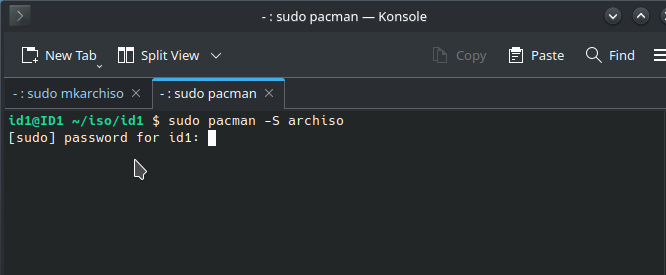
* **Installation d’Arch linux**

Nous avons utilisé le un guide d’installation disponible sur le site de Arch linux : https://wiki.archlinux.org/title/Installation\_guide

* **Installation du Packet Archiso**

Archiso est un outil extrêmement personnalisable pour la création d'images ISO de live CD/USB d'Arch Linux. Les images officielles sont construites avec Archiso. Il peut être utilisé comme base pour des systèmes de secours, des installateurs Linux ou d'autres systèmes.

Pour installer Archiso nous avons utilisé le gestionnaire de paquets **Pacman** d’Arch linux :



**Utilisation de Archiso pour la construction de la distribution personnalisée :**

* **Préparez un profil personnalisé**

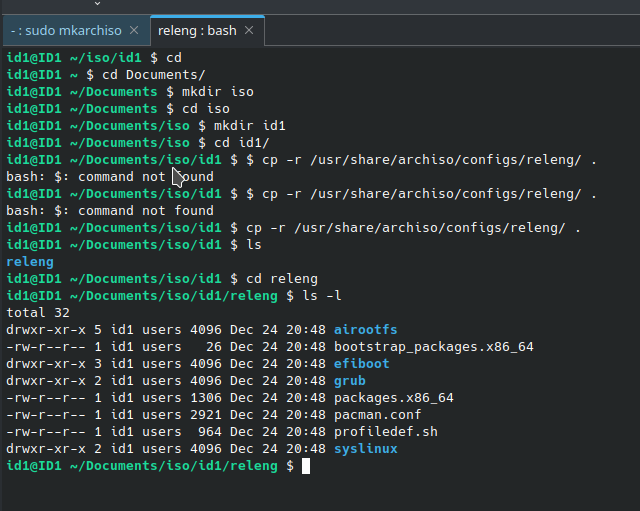
Archiso est livré avec deux profils, releng et baseline :

* **Releng** est utilisé pour créer l'ISO officiel d'installation mensuelle. Il peut être utilisé comme point de départ pour créer une image ISO personnalisée.
* **Baseline** est une configuration minimaliste qui inclut uniquement les paquets strictement nécessaires pour démarrer l'environnement live à partir du support.

Nous avons utilisé **releng.**

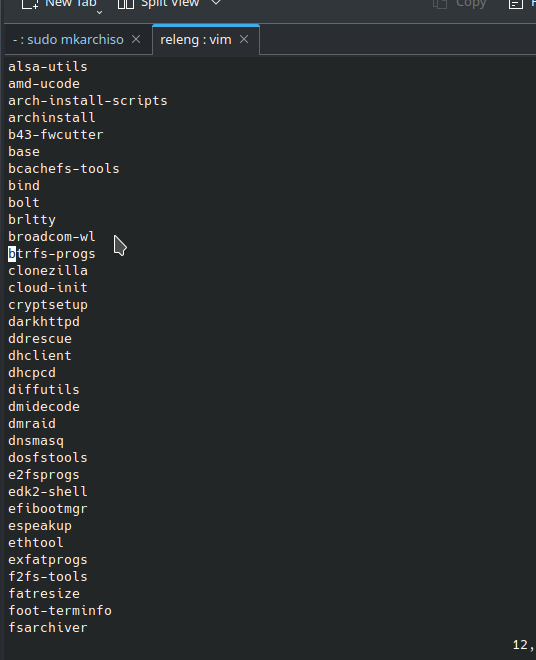
On va personnaliser l'un des profils inclus avec le paquet Archiso. Pour cela nous avons créé un dossier dans le lequel on va copier le profil **releng** de archiso

Ici le profil **releng** contient des fichiers qu’on va configurer pour la génération de l’image iso de notre distribution.



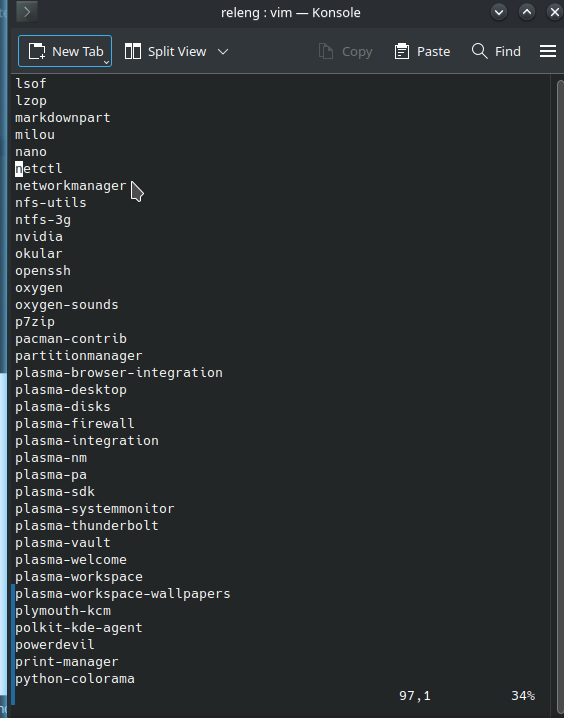
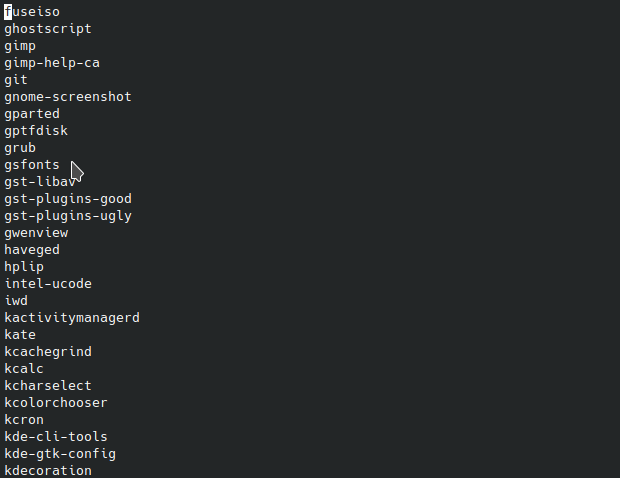
* **Le fichier packages.x86\_64**

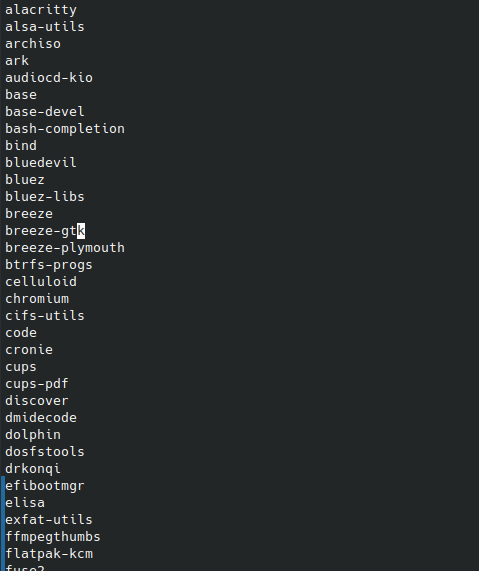
Ce fichier va par défaut contient les packages de nécessaires pour la construction d’une distribution basique qui sont listé ci-dessous :



On a ajouté ligne par ligne les packages qu’on veut qu’ils soient installé par défaut sur notre distribution :

On a inclus spécialement les packages pour le display manager qui **sddm ,** de l’environnement graphique **plasma** pour la configuration du réseau **networkmanager et network-manager-applet ,** le file manager **Dolphin** , un web browser pour la navigation sur internet **chromium** , des packages pour les périphériques d’entrées et sorties l’audio de vidéo et surtout les modules de python nécessaires pour l’intégration de notre systèmes de fichiers qui conçu exclusivement en python ce qui nous donne la liste des packages suivants :

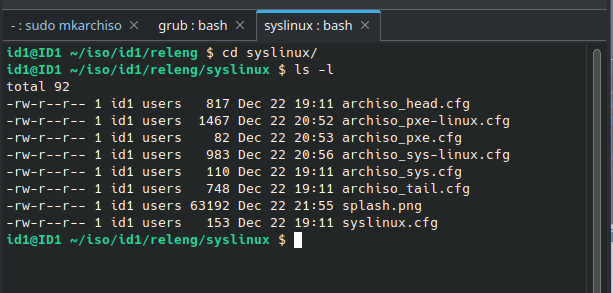


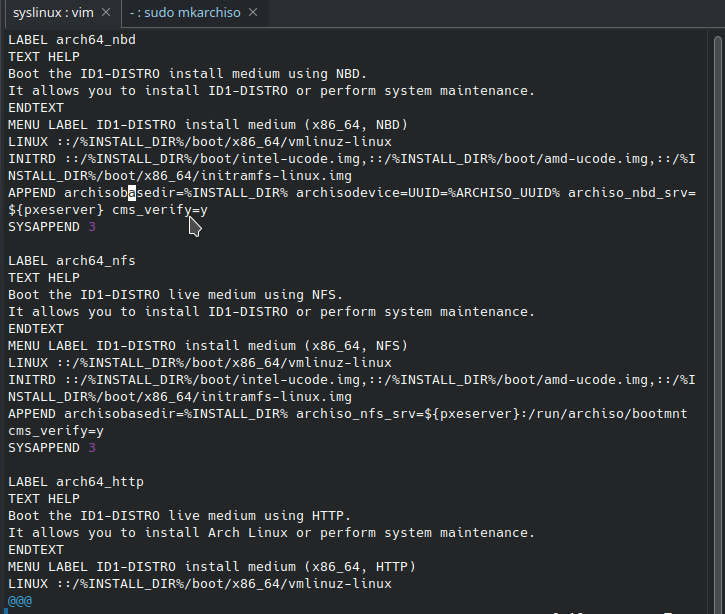
**Configuration de bootloader**

Par défaut archiso est configuré pour supporter le démarrage en BIOS à travers syslinux et pour les systemd à travers UEFI et GRUB

On a utilisé syslinux pour configurer le bootloader.

**Configuration du repértoire syslinux :**

****



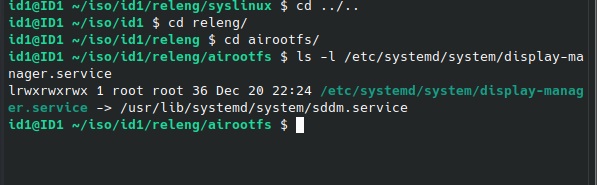
Ici on a changé l’image du background l’interface de l’arch linux lors de l’installation par notre propre :



**Intégration de l’environnement bureau (sddm et plasma) dans l’mage iso :**

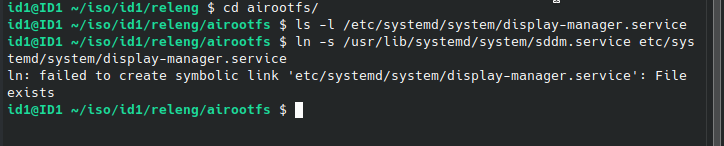
Le démarrage de X au démarrage est réalisé en activant le service systemd de votre gestionnaire de connexion. Pour le faire on va créer un lien symlink vers le service systemd du gestionnaire de connexion d’Arch linux installé sur l’hôte et juste utilisé le service de l’hôte :

Premièrement la commande suivante va nous permettre d’identifier l’emplacement du service systemd du gestionnaire de connexion du système hôte :



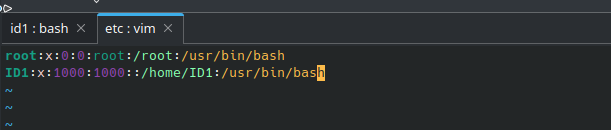
L’hôte utilise **sddm** et le packages existe dans la liste des packages à installer par défaut sur ID1-DISTRO.

Après on a créé le lien symbolique airootfs/etc/systemd/system pour **sddm**

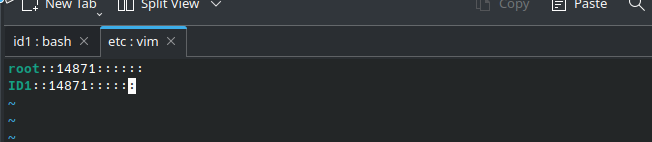


**Création et configuration des fichiers des utilisateurs:**

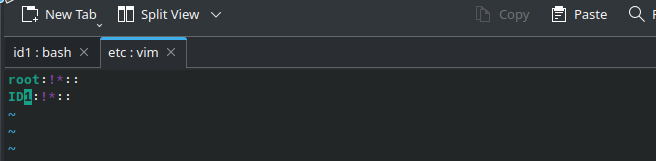
* **/etc/passwd**

 On a l’utilisateur **root** et un user simple pour l’installation :

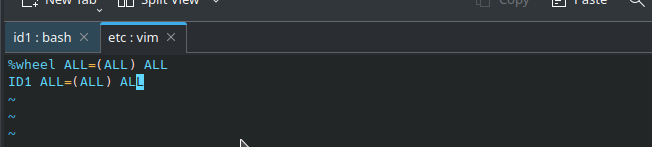
* **/etc/shadow**



* **/etc/gshadow**



* **/etc/sudoers**



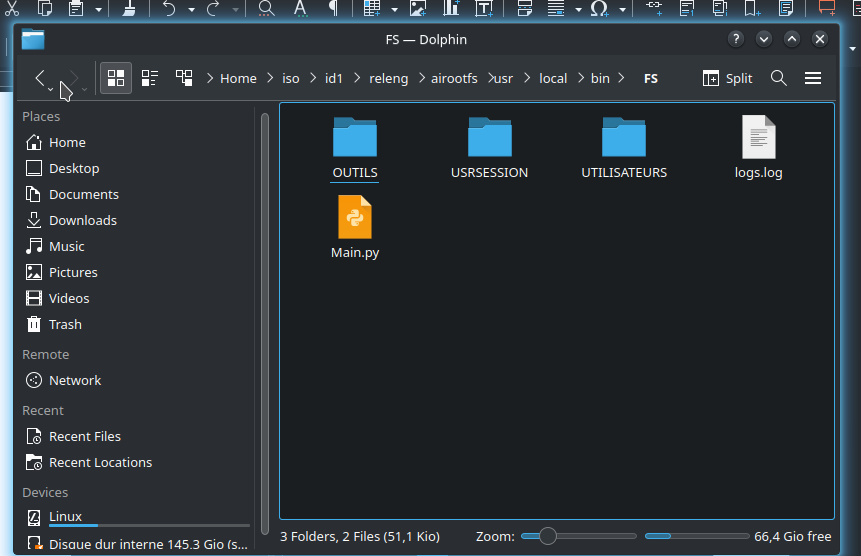
Ici on inclut l’utilisateur ID1 dans le fichier sudoers et après l’installation du système tous les utilisateurs créés appartenant au groupe wheel seront automatiquement inclus dans le fichier sudoers.

Maintenant nous allons intégrer le système de fichiers dans l’image iso.

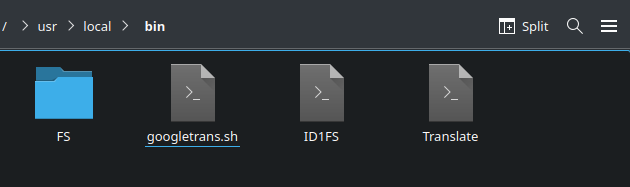
**3. Intégration du système de fichiers :**

* **Copie du système de fichiers dans le profil releng**

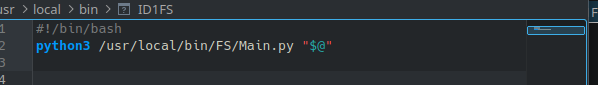
Ici on intégrer notre système de fichiers en copiant le dossier FS du script dans le répertoire /usr/local/bin pour qu’il soit accessible à chaque utilisateur créer dans la distribution :



Pour rendre accessible le système de fichiers de n’importe quel repértoire dans le terminal de la distribution, nous avons créé un script bash nommé ID1FS  dans ***/usr/local/bin***

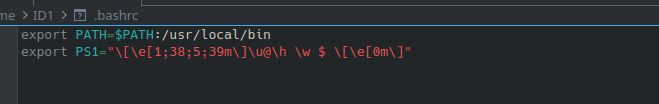


Voyons le contenu d’ID1FS :



le fichier ID1FS exécute le fichier Main.py du script principale de notre système de fichiers qui implémente toutes les fonctionnalités du système de fichiers.

Ceci dit on doit exécuter le script bash ID1FS dans le repertoire ***/usr/local/bin*** avant d’avoir accès au système de fichier donc pour pallier à ce problème on a inclus le repertoire ***/usr/local/bin*** la variable PATH du système en modifiant le fichier **.bashrc** comme suit :

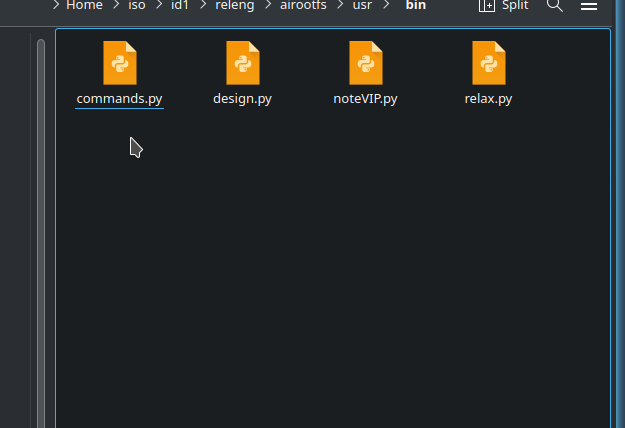


\*La deuxième ligne concerne juste la couleur du texte dans le terminal

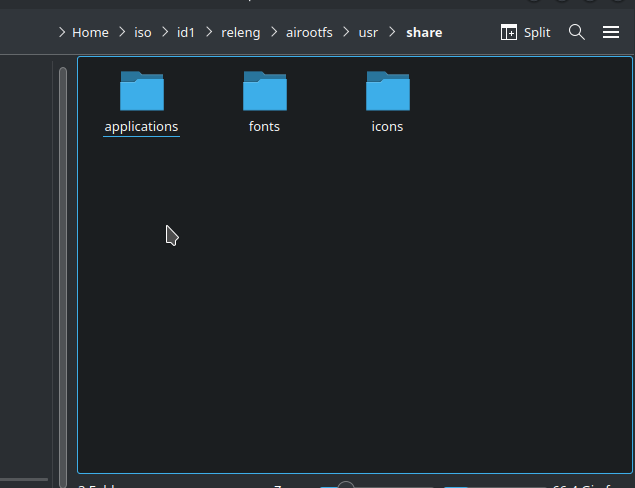
**4. Intégration des applications personnalisées :**

Chaque application a deux scripts un écrit en python qui constitue en quoi fait réellement l’application c’est à dire le backend et un fichier .dekstop qui permet à nos applications d’être visible dans le menu d’application.

Ici nous avons décidé de mettre les scripts gérant le background dans ***/usr/bin***

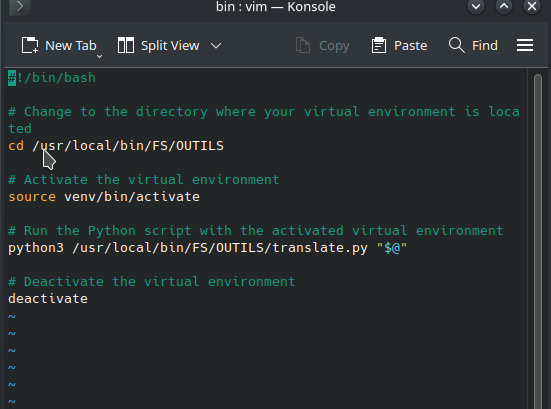


En ce qui concerne les fichiers .desktop et aussi les logos fonts utilisés par les applications les a mis dans le repertoire ***/usr/share***



**5. Intégration du programme Translate :**

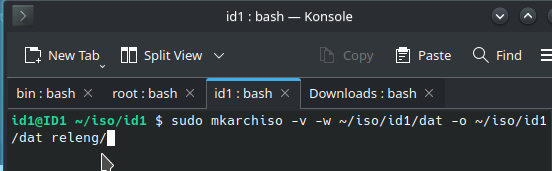
Le programme Translate est un script python qui permet de faire la traduction d’un langage à un autre sans besoin d’accéder a un browser. Pour son intégration, on crée un script shell qui permet d’exécuter le programme de n’importe quel répertoire de la distribution :

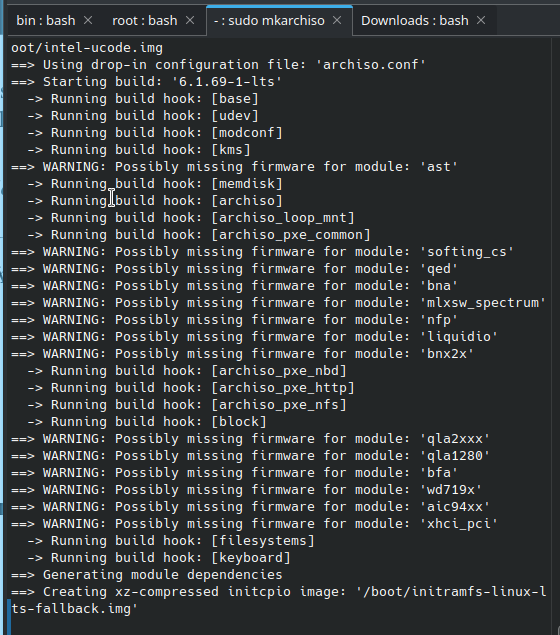


\* on a utilisé une environnement python pour cette commande car il nécessite des bibliothèques comme googletrans qui ne sont pas disponible dans Arch linux

**6. Création de l’image ISO :**

Pour la création de l’image iso avec **archiso** de notre distribution on a utilisé la commande suivante :





**Conclusion :**

*En conclusion, la réussite du développement et de l'intégration du système de fichiers dans la distribution Linux personnalisée constitue une réalisation majeure. Ce projet a été l'occasion d'appliquer des compétences avancées en programmation système et en administration Linux, tout en offrant une opportunité d'acquérir de nouvelles connaissances. Plus qu'une simple exploration technique, cette expérience a également permis de développer des compétences essentielles en travail d'équipe dans le contexte d'un projet informatique. L'échange d'idées, la collaboration et la résolution conjointe de défis ont contribué à renforcer l'esprit de travail en groupe, enrichissant ainsi notre expérience académique et professionnelle.*

**Remerciement :**

*Nous tenons à exprimer notre profonde gratitude au Professeur* ***Cherradi Mohamed*** *pour avoir proposé et encadré ce projet stimulant. Sa vision éclairée et son encadrement attentif ont été des catalyseurs essentiels de notre réussite. Nous sommes reconnaissants pour l'opportunité d'explorer le développement d'un système de fichiers et d'une distribution Linux personnalisée sous sa direction. Ces enseignements précieux resteront ancrés dans notre parcours académique. Merci infiniment, Professeur* ***Cherradi Mohamed****.*