**Rapport de Projet IHM Python**

# Introduction

Le but de ce projet est de faire une application Homme Machine en utilisant le langage de programmation python. Le but de cette application est de faire la configuration des routeurs facilement pour l’utilisateur ou un client simple qui ne connait pas comment configurer les routeurs manuellement avec des codes linux.

J’ai utilisé pour crée cette application le Qt Designer pour faire la partie designer ou autrement dit la partie front end que l’utilisateur peut le voir et aussi j’ai utilisé le PyQt5 pour créer la partie back end et toutes les fonctionnalités à taper. Afin de pouvoir connecter mon interface au routeur, j’ai utilisé le protocole SSH, c’est un Secure Shell qui peut contrôler une machine distante en toute sécurité. J’ai trouvé ce projet très intéressant et enrichissant. Cependant, j’aurais aimé avoi un peu plus de temps afin d’augmenter ses fonctionnalités comme par exemple lier l’IHM avec une base de données.

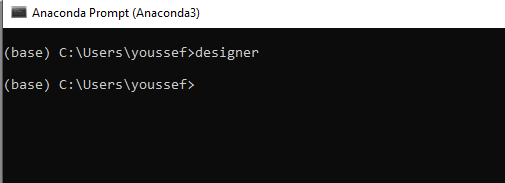
# Remerciement

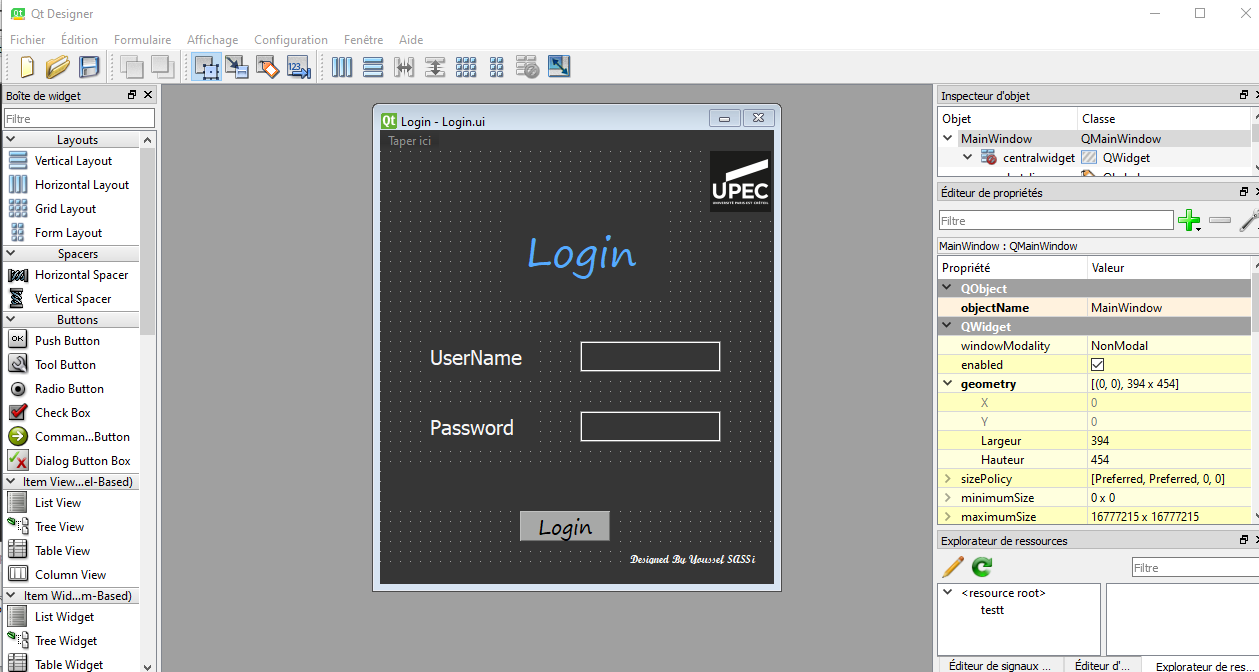
Tout d’abord j’aimerais remercier mon professeur M. Thiago Abreu pour son encadrement et son orientation dans ce travail tout au long de ce projet. Je remercie également tous mes enseignants qui ont su nous apporter le temps, la confiance et les explications qui m’ont été utiles pour la réalisation de ce projet.

Finalement, je remercie l’école, l’Episen pour la bonne formation et les conditions favorables d’étude.

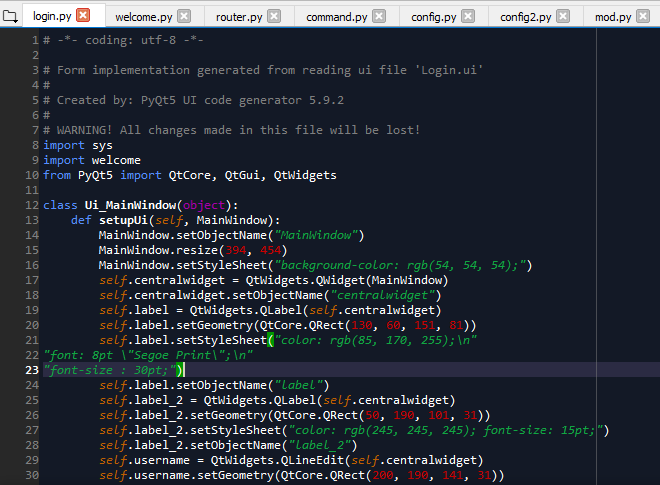
# IHM avec QT Designer

## Fenetre 1





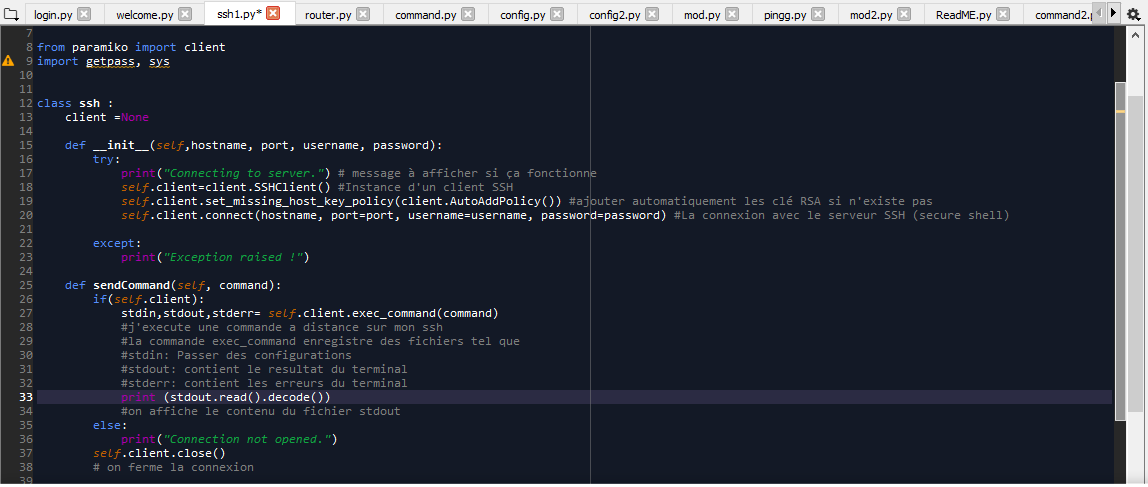
J’ai utiliser le Qt Designer pour former interfaces de travail , et pour commencer il faut lancer anacondat prompt et écrire « Designer » , après de terminer tous les interfaces et après d’ajouter les boutton ,les labels , les textline etc.. j’ai convertir tous les fichier UI en Py par la commande «  pyuic5 -x nom\_fichier.ui -o nom\_fichier.py »





On peut maintenant trouver notre interfaces en code python pour faire les fonctionnalités et aussi tous les modifications nécessaire (pour passer à la partie back end )

## Connection SSH



Le but de SSH est d’accéder à une machine virtuelle ( le routeur dans notre cas) via python pour faire les configuration facilement pour un client ou un utilisateur quelconque. Donc la librairie Python la plus adaptée à l’utilisation du protocole SSH, en tant que client ou serveur, se nomme **Paramiko**. Se servir de cette librairie impose de prendre en compte des spécificités de dialogues des interlocuteurs. Et aussi dans le but de lancer une ou des commandes Linux sur une serveur distant avec SSH via python facilement.

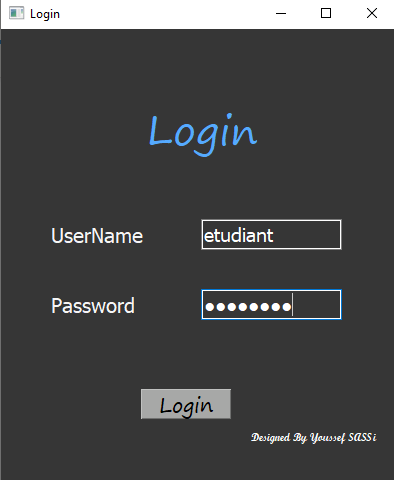
Autrement dit le SSH c’est comme **un tunnel** entre l’application python que j’ai faite et le routeur ( Machine virtuelle )

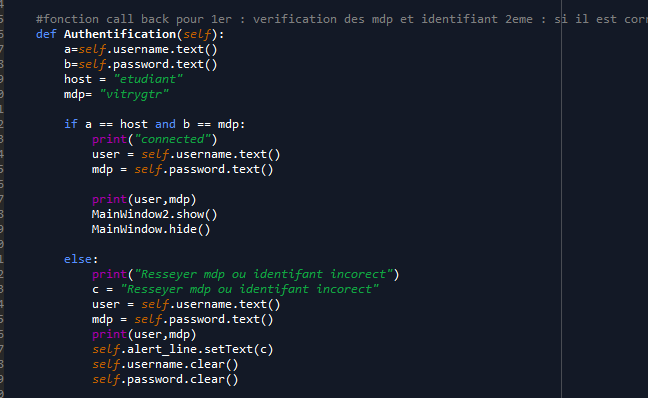
Dans la classe SSH, on définit les variables stdin, stout et stderr qui correspondant respectivement aux input, output et message d’erreur de Python, ces variables sont en fait le résultat des commandes passées en langage Linux. D’autant plus, la méthode sendCommand permet de renvoyer la sortie d’une commande Linux.

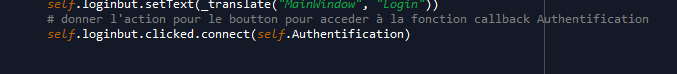
## Les Fonctionnalités

### 3.3. Les interfaces

#### **Login.py**



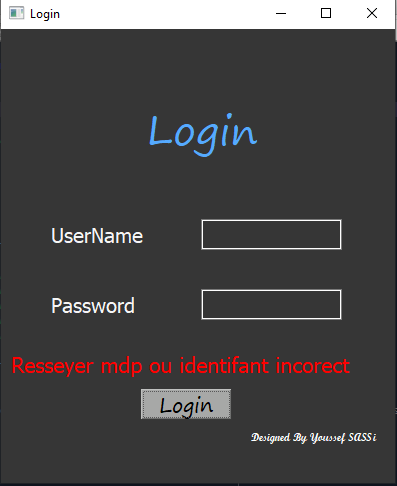


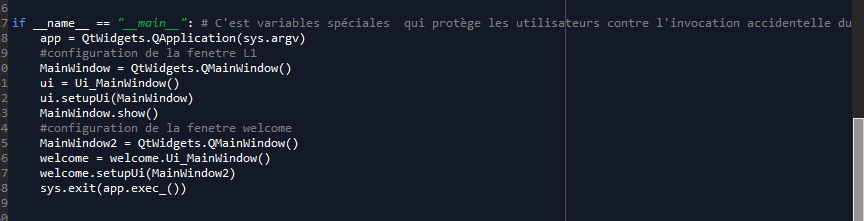


J’ai crée une fonction callback Authentification liée avec le button « loginbut » j’ai donnée l’action çe boutton pour executé la fonction

J’ai fait une condition if pour les vérifications pour l’identifiant et le mot de passe si c’est correct il va passer à la deuxième fenetre sionon il va afficher un message d’erreur et aussi il va effacer les champs dans les « Editline »

J’ai identifier les getter ( les champs des texts ) avec ses objectname



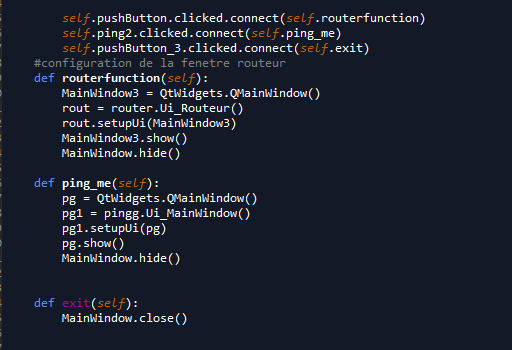


Pour passer de fenetre à l’autre il faut déclarer premièrement la fenetre suivant dans comme un objet et n’oublie pas d’importer le nom de fichier de fenetre dans l’interface ici

### **Welcome.py**



J’ai mit dans cette interface 3 boutton , une pour quitter l’application , la deuxième pour passer au interfaces de choix de routeur et la troisième c’est pour le ping : pour tester la connexion



J’ai fais cette fois 3 fonction callback :

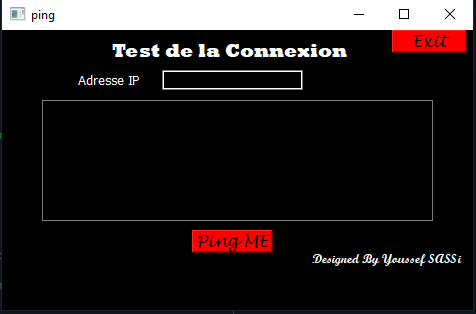
1er : pour passer à la fenetre de routeur

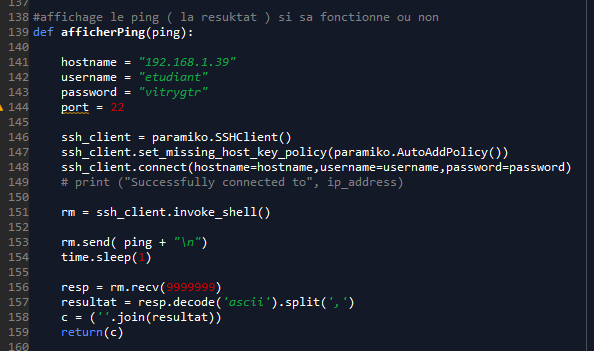
2eme : pour le ping : le test de connexion

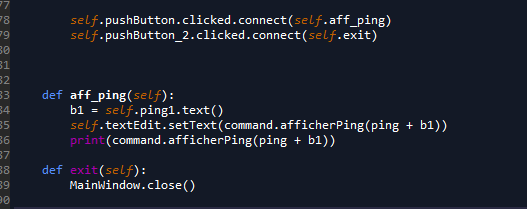
3eme : pour quitter le programme

Et j’ai les relier avec des boutons pour faire les actions

### **Pingg.py**



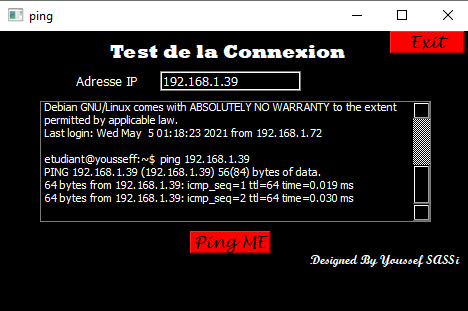




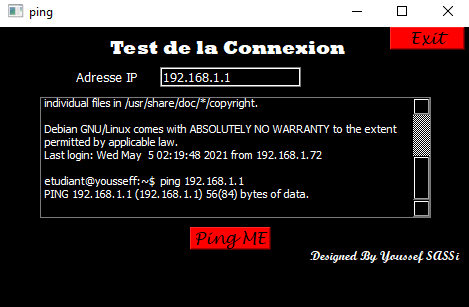
J’ai fait une fonction afficherPing dans le fichier command ou j’ai mit tous mes commandes de travail et après j’ai fait une autre fonction callback aff\_ping : j’ai mit la resultat de output de terminal dans le setteur de textedit ou il va afficher la resultat de ping et aussi j’ai fait une print sur le console pour vérifier si ça marche ou non

J’ai utilisé invock\_shell() pour Demander une session shell interactive sur ce canalrt si le serveur le permet, le canal sera alors directement connecté aux stdin, stdout et stderr du shell. Et je l’ai mit dans un variable pour envoyé la requet ou la commande ( linux) après j’ai fait une pause de durré une secande pour que la commande s’execute car le ping affiche la resultat 64 byte en cas de succées après avec recv il va retourné la resultat en chaine de caractere avec avec un quantité maximale de valeur

**Dans le cas de succès**



**Dans le cas d’échec ( pas de connexion)**



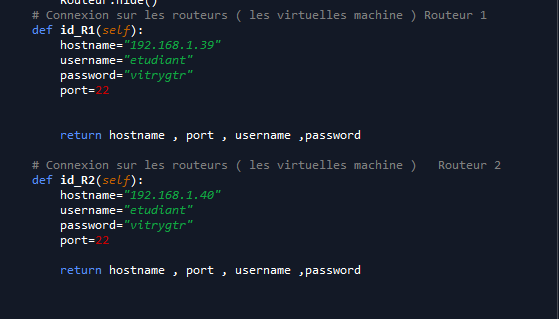
### **Router.py**





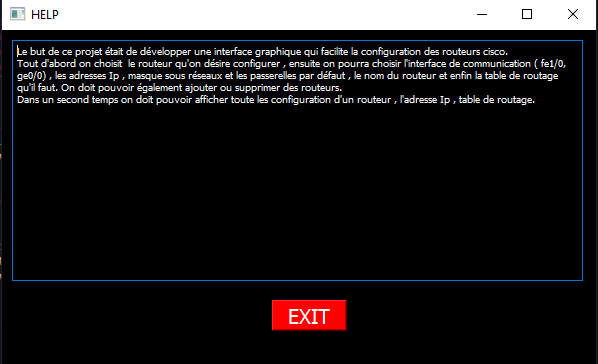
Dans cette interface on va trouver 3 boutton :

Deux boutton pour les choix des routeurs pour les configurer ; dans cette partie j’ai fait deux fonction callback lier avec des boutton pour passer à les fenêtres choisi config pour le routeur 1 et config2 pour le routeur 2 , ainsi un boutton READ Me pour passer à l’interface ou j’ai mit les consignes et comment utiliser l’application pour l’utilisateur

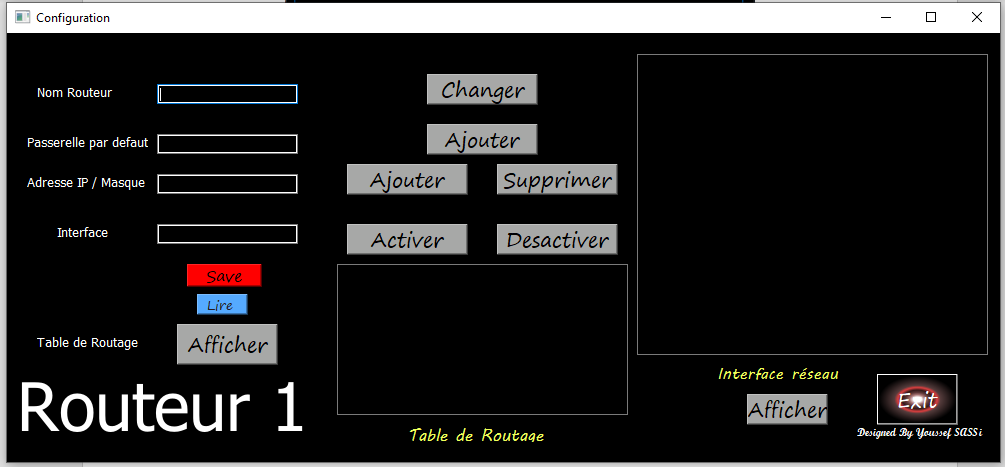


La connexion pour accéder si pour le routeur 1 ou routeur 2 (virtuel machine ) avec des adresse IP static l’utilisateur peut l’accéder et connecter par SSH

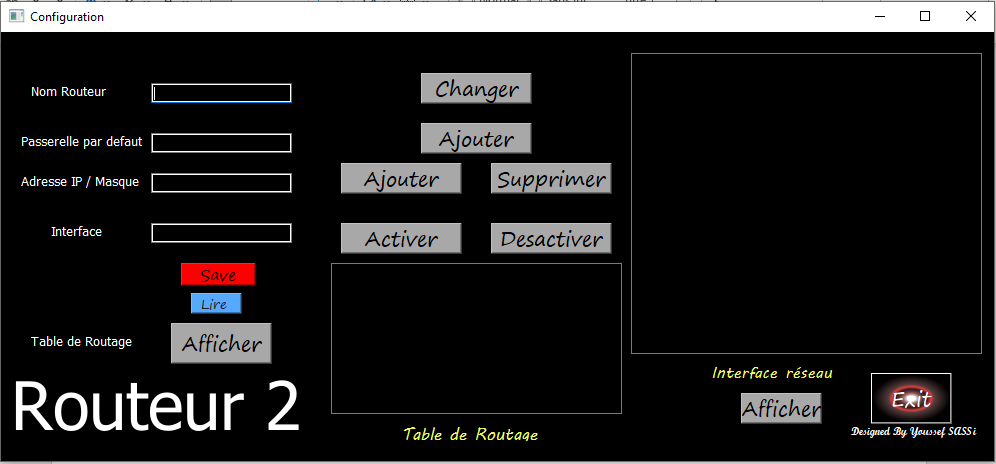
### **ReadME.py**



### **Config.py**



### **Config2.py**



Cette interface de configuration de routeur est le plus importante dans toute l’application

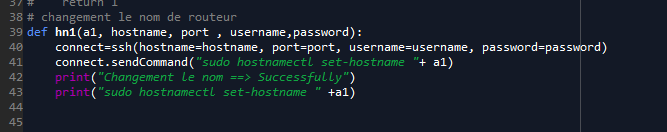
On trouve des case (edit\_line) pour remplire les information, des zone de text pour afficher les resultat aussi des boutton pour sauvgarder , lire, afficher , changer , ajouter , supprimer , activer , désactiver relier avec des fonctions API

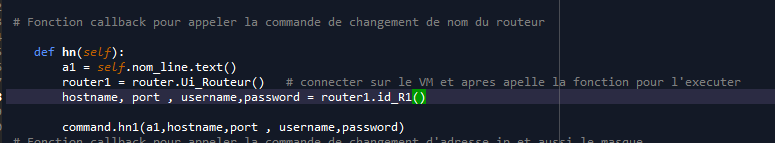
Pour changer le nom de routeur (hostname) :

J’ai créé une fonction dans la fichier command.py pour pour être facile à l’accéder à tout moment et facile à l’utiliser

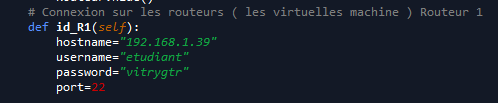
J’ai déclaré pour cette fonction une variable a1 on j’ai mit le getteur de la zone ou je vais écrire le nom de routeur

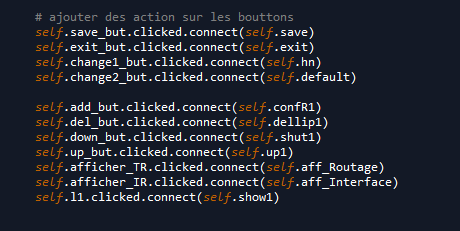






Connexion de ssh sur le fichier Router.py

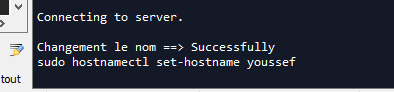




Donc dans un premier temps j’ai crée une fonction de connections sh avec l’injection de la commande de linux pour kil envoie ( sendCommand ) « Stdrin »

Dans la deuxième j’ai fait une fonction callback « hn » ou il prend l’acces pour connecter sur le ssh par le test avec hostname port username password qui sont declaré dans la fonction id\_R1 dans le fichier router.py

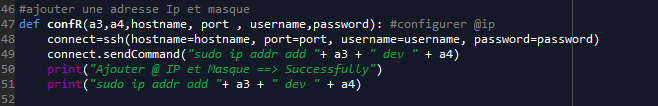
Et dans la 3eme on va appeler la fonction hn de la ficher command.py pour l’executé dans le boutton change1\_but

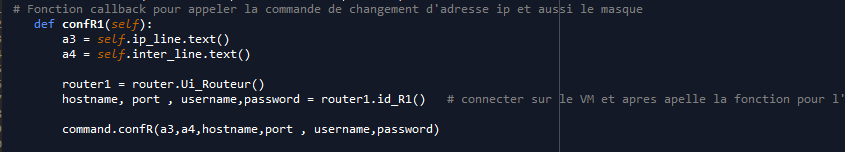


Ajouter et supprimer une adresse ip et masque :

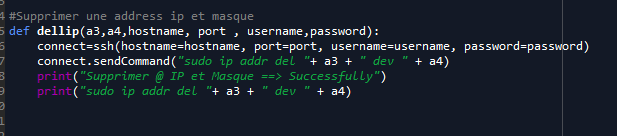


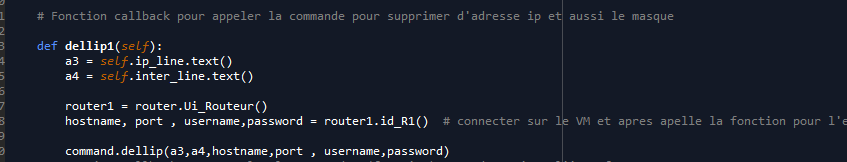
**Ajouter :**





**Supprimer :**



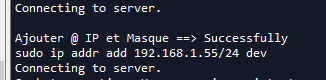


Même idée et même fonctionnalité comme le changement de hostname mais cette fois on change la commande linux par la commande d’ajouter une adresse ip et aussi on peut supprimer

**Demo :**

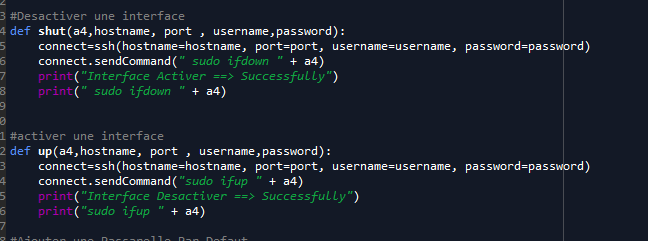
**Ajouter**



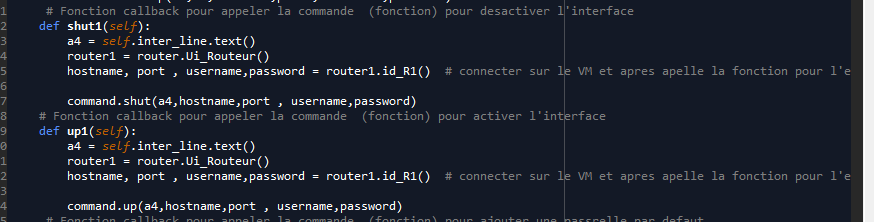


Activer et désactiver les interfaces :

Partie command.py

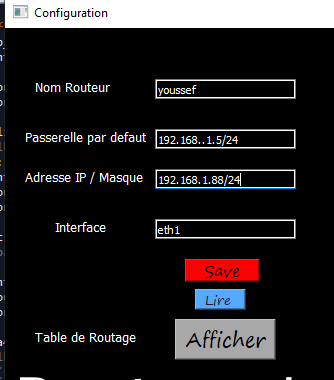


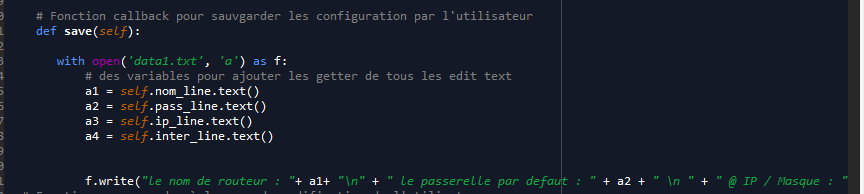
Partie config.py

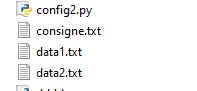


Même fonctionnalités et même idée pour activer et désactiver l’interface choisi par l’utilisateur soit eth0 , soit eth1 . des fonction callback relier avec des boutton dans le fichier config.py

**Enregistrement :**



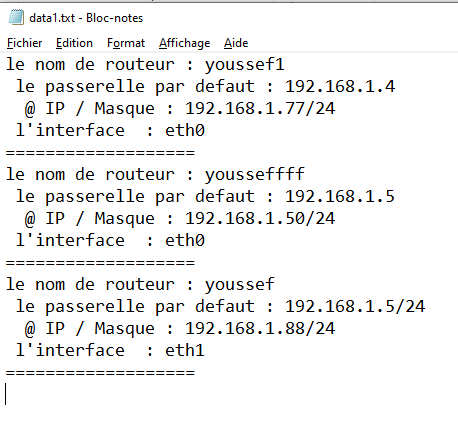




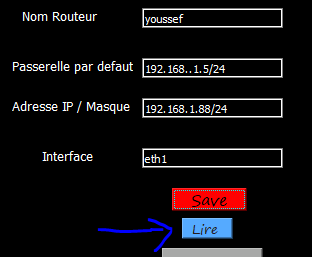
J’ai mit une fonction d’enregistrement txt lier avec le boutton « save » j’ai crée un fichier data1.txt pour l’interface config pour routeur 1 avec le nom data1.txt et un autre fichier pour le routeur 2 avec le nom data2.txt

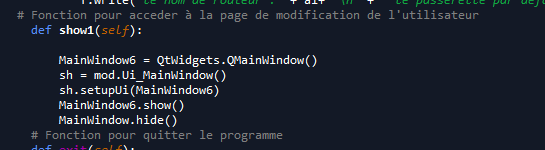
Il va ouvrir le fichier pour ajouter ou stocker les donner sans ecrasser les données precidents avec 🡺 « a » il va enregistrer tous les modification que l’utilisateur va le faire depuis son configuration sur les champs des text.

Exemple :

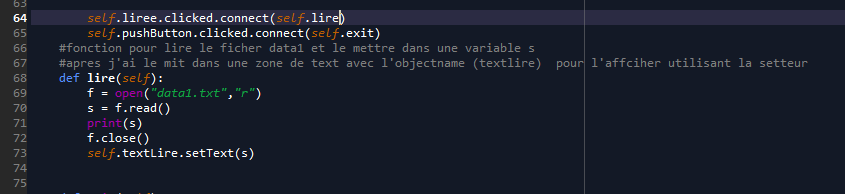


### **Mod.py**









Dans cette fonction même principe de sauvegarde mais cette fois on va lire les donner d’un fichier txt

Avec « r » read et j’ai mis dans un variable « s » et j’ai le mit dans le setteur de la zone de text pour l’afficher à l’utilisateur

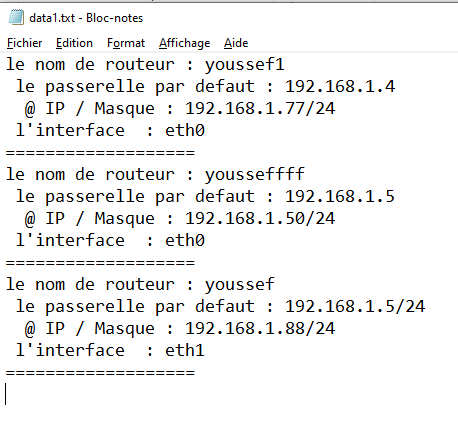
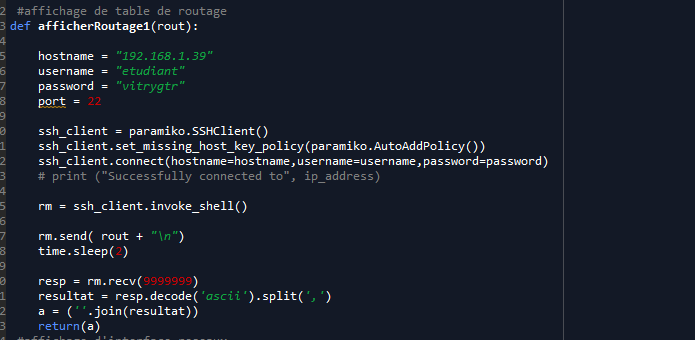


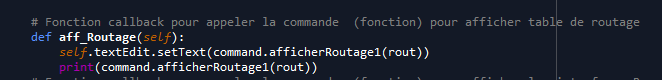
Table de routage , j’ai fait une fonction

Partie command

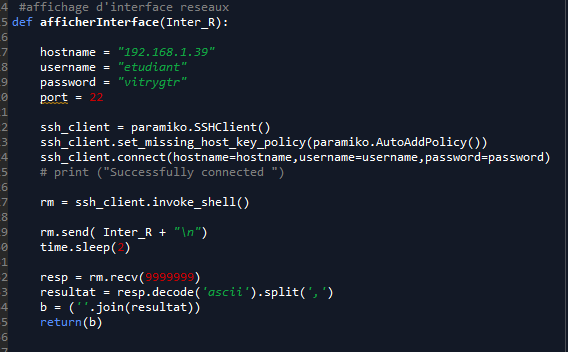
On va envoyer stdrin pour exécuter la requet et après on recuper les donner comme chaine de caractere pour le mettre dans un variable injecter au setteur de la zone de text pour l’afficher ç l’utilisateur

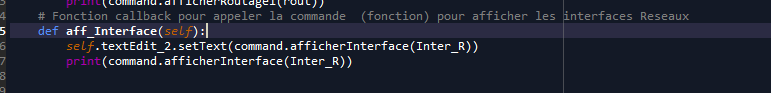


Partie config : la fonction callback pour appeler la fonction précédente et le mettre dans un setteur pour l’afficher , affichage pour utilisateur et affichage print sur le console pour moi-même pour vérifier si ça fonctionne correctement ou non



**Affichage des interfaces réseaux ( ip a )**





meme fonctionnalité à la fonction de table de routage