

DOCUMENTATION DU PROJET DE RESEAUX ET PROTOCOLE SUR LE THEME :

**DEVELOPPEMENT D'UN PROTOCOLE
ET D'UN SERVEUR DE FICHIERS
POUR TELECHARGEMENT DE
FICHIERS.**

Nom et Prenom Etudiants :

-KOFFI YEBOUA BADOU GRACE E.
-TRAORE ELIE

Nom Professeur :

Dr DIALLO MOHAMED

SOMMAIRE

Introduction

-Présentation et objectifs du projet.

I-Analyse des besoins.

- 1.Description des besoins du projet.
- 2.Études de faisabilité.
- 3.Exigences techniques et fonctionnelles.
- 4.Spécifications détaillées.

II.Architecture du système.

- 1-Schéma de l'architecture du système
- 2.Composant du système.
- 3.Sécurité du système et mécanisme de gestion des erreurs.

VImplémentation du système.

- 1.Développement du serveur de fichiers.
- 2.Tests et validation du système.
- 3.Installation et utilisation du système.

Conclusion

- 1.Bilan du projet.
- 2.Perspectives d'évolution et limite du système.
3. référence et liens github du projet.

Introduction

Un réseau est un ensemble d'ordinateur et d'autres périphériques qui sont connectés entre eux pour échanger des données ou des ressources. Il peut être utilisé pour diverses applications telles que la communication, l'accès à internet et le transfert de fichiers. Pour ce faire, un ensemble de règle et de procédure sont mis en place : ce sont des protocoles. Dans le cadre de notre projet, nous serons amenés à mettre en place un protocole et un serveur de fichiers qui permettront à des clients de télécharger des fichiers. Dans notre cas, le protocole sera utilisé pour faciliter la communication entre les clients et les serveurs. Ce projet réalisé à la demande de Docteur Diallo enseignant à l'Université Félix Houphouet Boigny de Cocody sera mis en œuvre par une équipe de deux étudiants qui sont : Traoré Élie et Koffi Yéboua Badou Grace Evelyn.

Présentation et objectifs du projet.

Le but de notre projet sera de fournir une solution efficace et sécurisé pour le téléchargement de fichiers par plusieurs clients. Le serveur de fichiers sera responsable de stocker et de gérer les fichiers, tandis que le protocole sera utilisé pour faciliter la communication entre le client et le serveur. La mise en place de cette solution implique plusieurs étapes, y compris la définition des exigences du projet, la conception de l'architecture du système, le choix des technologies appropriées, l'implémentation, les tests et la documentation.

Dans cette documentation,nous décrirons les différentes étapes impliquées dans la mise en place de ce système en passant en revue les choix d'architectures ,de technologies et d'implémentations.Nous couvrirons également les tests effectués sur le système .Nous présenterons aussi les limites de notre système et les difficultés rencontrées pour sa conception.

Nous espérons que la mise en place de cette solution permettra d'améliorer considérablement le partage de fichiers entre clients et serveur .

I-Analyse des besoins.

1.Description des besoins du projet.

Dans le cadre de notre projet de mise en place d'un protocole et d'un serveur de fichiers nous aurons besoins de mettre en place :

- Une interface client pour permettre aux clients de sélectionner et de télécharger des fichiers.
- Un système de gestion de fichiers pour stocker et organiser les fichiers téléchargés .
- Un protocole de communication pour assurer une transmission de donnée fiable et sécurisé entre le serveur et les clients.
- Un système de notification pour informer les utilisateurs du succès ou de l'échec de leur téléchargements.
- Un système de journalisation pour enregistrer les activités de téléchargement et de connexion des utilisateurs,afin de faciliter le dépannage et la surveillance du serveur.

2.Études de faisabilité.

-Évaluation des compétences : Pour ce projet , les compétences techniques nécessaires incluent la programmation,la gestion de serveurs et la sécurité informatique.Il nous faut avoir des compétences approfondie en python, java ou Go.Il est également important que nous comprenions les protocoles de communication tels que TCP|IP,HTTP etc.....

-Évaluation des coûts : Ce projet ne nous coûtera quasiment rien dans la mesure ou tous les outils que nous allons utiliser sont open source.

-Évaluation de la scalabilité : Bien que ce projet soit de petite envergure,il est toujours important de tenir compte de la scalabilité pour répondre à une demande croissante d'utilisateurs.

-Évaluation des avantages : la mise en place d'un serveur de téléchargement de fichiers peut permettre d'acquérir de précieuses compétences techniques ,telles que la programmation,la gestion de serveurs .

3.Exigences techniques et fonctionnelles.

Exigences techniques

-Le serveur doit pouvoir gérer les connexions entrantes et sortantes en utilisant le protocole de communication TCP|IP.

-Le serveur doit être capable de stocker des fichiers de différents types et tailles.

-Le serveur doit être capable de gérer les erreurs de connexion et de transfert de fichiers.

-Le serveur doit pouvoir gérer plusieurs demandes de téléchargement simultanément sans affecter les performances globales.

-Le serveur doit pouvoir gérer la connexion simultanée de plusieurs clients.

Exigences fonctionnelles

-Les clients doivent pouvoir télécharger des fichiers de différents type et taille en utilisant une interface utilisateur simple et conviviale.

-Les clients doivent pouvoir voir la liste des fichiers disponibles sur le serveur.

-Les clients doivent pouvoir créer des fichiers dans le dossier de fichiers du serveur.

-Le serveur doit permettre aux utilisateurs de supprimer des fichiers de son dossier de fichier.

-Le serveur doit permettre aux utilisateurs de se déconnecter du serveur.

3. Spécification détaillée.

-Le système doit permettre aux clients de télécharger des fichiers depuis le serveur en utilisant une interface dédié.

-Le système doit être capable de stocker les fichiers de manière sécurisé sur local.

-Le système doit utiliser le protocole de communication TCP|IP pour garantir une communication fiable et sécurisé entre les clients et le serveur.

-Le système doit prendre en charge plusieurs connexions simultanées pour permettre à plusieurs clients de se connecter simultanément.

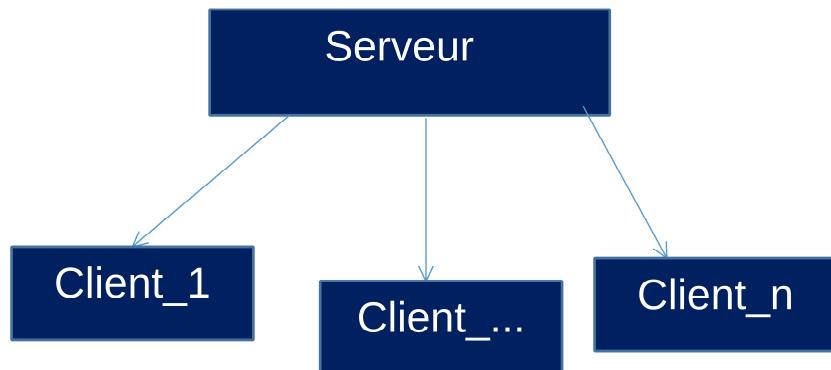
-Le système doit gérer les erreurs de connexion et de transfert de fichiers.

-Plusieurs clients peuvent télécharger le même fichier simultanément.

II. Architecture du système.

1-Architecture et composants du système.

-Architecture du système :



-Composants du système :

-Un serveur local

-Un composant de gestion de transfert de fichiers qui gère le transfert de fichiers entre le serveur et les utilisateurs.

Pour ce projet ,nous utiliserons le langage de programmation python , le protocole TCP|IP pour la communication entre le client et le serveur .

3.Sécurité du système et mécanisme de gestion des erreurs.

Dans le cadre de ce projet,nous aurons à gérer les erreurs suivantes :

-Dans le cas où le serveur ne contient aucun fichier le message suivant sera affiche « SERVER DIRECTORY EMPTY»

-Dans le cas où le client souhaite télécharger sur le serveur un fichier inexistant , le message suivant sera affiche « **ERROR 401 :FILE 'nom_fichier' NOT FOUND ON SERVER**»

-Dans le cas ou le téléchargement d'un fichier a échoué un message de type « **ERROR 402 : DOWNLOAD FAILED** ».

-Si la connexion au serveur a réussi le message : « **200 : CONNECTED SUCCESSFULL** » sera affiché.

-Si un client se déconnecte du serveur le message « **200 : DISCONNECTED SUCCESSFULL**».

-Si le client souhaite faire autre chose que ce que le système est capable de faire le message « **ERROR 403 : Invalid choice tape HELP for see all command** ».

V.Implémentation du système.

1.Développement du serveur de fichiers.

Fonctionnalité du serveur :

-Connections et déconnexions simultanées de plusieurs clients.

-Fournir la liste des fichiers disponibles dans son répertoire de fichiers.

-Fournir le fichier demandée par le client pour téléchargement.

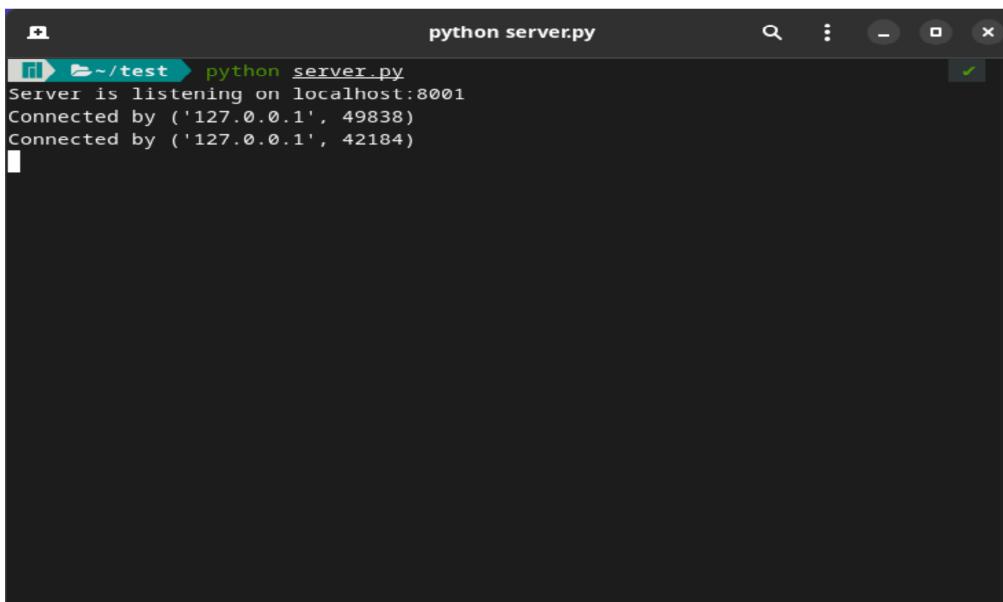
-Permettre au clients de créer des fichiers dans son répertoire de fichiers.

-Permettre aux clients de supprimer des fichiers de son répertoire de fichiers.

Interface :

-Interface serveur :

L'interface de notre serveur se présente comme suit(cette interface peut être différentes en fonction de votre système d'exploitation) :



```
python server.py
Server is listening on localhost:8001
Connected by ('127.0.0.1', 49838)
Connected by ('127.0.0.1', 42184)
```

-Interface client :

Notre interface client se présente comme suit(cet interface peut être différentes en fonction de votre système d'exploitation) :

```
python client1.py

Enter the filename to download: fic
ERROR 402 : DOWNLOAD FAILED

ERROR 401: FILE fic NOT FOUND ON SERVER

>>>>> DEL

Enter the filename to delete: fic
ERROR 401: FILE fic NOT FOUND ON SERVER

>>>>> NEW

Enter the filename to create: fic

File fic created on server

>>>>> LIST

Files on server:
fic

>>>>>
```

Choix de logiciel et normes de codage du logiciel

Pour réaliser ce projet nous avions le choix entre trois langages de programmation qui sont : Python ,Java et Go.Nous avons opté pour le langage python parce que c'est un langage assez simple que nous avons utilisé depuis la première année,c'est un langage très riche avec de nombreuses facettes.Aussi vu les contraintes de temps qui nous ont été impartis ce langage nous permettrait de respecter les délais.Nous utiliserons les bibliothèques suivantes :

-threading

-socket

-os

Concernant les normes de codage ,nous avons décidé d'utiliser la programmation modulaire c'est à dire utiliser des fonctions ou des procédures pour implémenter chaque fonctionnalité ce qui assurera la maintenabilité et la ré-utilisabilité de ce code. Pour le nommage des

variables nous avons utilisé les règles de nommage standards(pas d'espace dans le nom des variables, la première lettre du nom de la variable doit être une lettre alphabétique , l'utilisation des accents est interdit etc.....)

2.Tests et validation du système.

Fonctionnalités	Résultats attendus	Résultats obtenus
Connexion de plusieurs client.	200 : CONNECTED SUCCESSFUL pour chaque client.	200 : CONNECTED SUCCESSFUL pour chaque client.
Téléchargement de fichiers	File {nom_fichier} downloaded from server .	File {nom_fichier} downloaded from server .
Création de fichiers	File{nom_fichier} created on server	File{nom_fichier} created on server
Suppression de fichiers	File{nom_fichier} deleted from server	File{nom_fichier} deleted from server
Déconnexion	200 : DISCONNECTED SUCCESSFULL	200 : DISCONNECTED SUCCESSFULL
Affichage de la liste des fichiers.	Affichage de la liste des fichier dans le dossier nomme server_directory	Affichage de la liste des fichier dans le dossier nomme server_directory

3.Installation et utilisation du système.

Pour utiliser notre système il faut juste

-installer le langage python(version3) sur sa machine .

-Une fois python installer entrer dans l'invite de commande ou le terminal(taper cmd dans la barre de recherche pour les utilisateurs de windows,pour les utilisateurs de linux cela pourrais dépendre de votre distribution linux pour UBUNTU taper tilix dans la barre de recherche,pour manjaro le terminal se trouve juste en bas dans la barre des applications etc....)

-Assurer vous de vous trouver exactement dans le répertoire où se trouve le dossier contenant le code source du serveur , du client et du dossier de fichiers du serveur(dans notre cas ce dossier se nomme server_directory).

-Démarrer le serveur : Entrer le mot clé **python** suivie de server.py(python server.py) et le serveur sera en marche.

-Connecter vous maintenant au serveur en ouvrant autant de session que vous souhaitez(ouvrir un autre terminal et entrer python client.py tout en s'assurant d'être dans le bon répertoire.

-Entrer la commande **HELP** pour voir la liste de toutes les commandes disponibles.

-Une fois terminer déconnectez vous ,vous pouvez laisser le serveur en marche autant de temps que vous souhaitez ou le déconnecter à son tour.

COMMANDE

Notre système dispose de 6 commandes qui sont :

-La commande **HELP** permet d'afficher la liste de toutes les commandes disponibles sur le système.

- La commande **LIST** permet d'afficher la liste de tous les fichiers disponibles dans le dossier de fichiers du serveur.
- La commande **DOWNLOAD** permet de télécharger un fichier dans le dossier de fichiers du serveur.
- La commande **DEL** permet de supprimer un fichier dans le dossier de fichiers du serveur.
- La commande **NEW** permet de créer un fichier dans le dossier de fichier du serveur.
- La commande **DISCONNECT** permet de se déconnecter du serveur.

Conclusion

1.Bilan du projet.

Dans cette documentation,nous avons essayé de présenter la définition des exigences du projet ,la conception de l'architecture du système ,le choix des technologies appropriées ,l'implémentation et les tests.Nous avons opté dans ce projet pour un système simple ,facile d'utilisation avec des fonctionnalités additionnelles.Tout en espérant que ce système réponde aux attentes des différents utilisateurs nous savons qu'il comporte quelques limites et peut être améliorer pour un meilleur usage.

2.limites du système et perspectives d'évolution .

-Limites du système :

Bien qu'étant fonctionnelle notre système a les limites suivantes :

-Un client ne peut télécharger,supprimer ou créer qu'un et seul fichier à la fois par exemple si le client souhaite télécharger,supprimer ou

créer deux fichiers il ne pourra pas le faire dans une seule requête ou commande.

-Après le téléchargement du fichier le client est obligé de se déconnecter(en faisant ctrl C) et se reconnecter pour au serveur pour faire d'autre requête . Ceci est un défaut de conception.

-Ce système fonctionne uniquement pour un serveur local , il ne peut être utilisé avec un serveur distant ou un serveur web.Donc nous ne pouvons pas effectuer des requêtes pour obtenir des pages .HTML.

-Ce système permet aux clients de télécharger tout type de fichier (fichier texte,fichier pdf ,fichier image) cependant pour les fichiers image il faudrait attendre quelque seconde après le téléchargement pour pouvoir regarder l'image téléchargée.

-Le client ne peut pas télécharger des dossiers de fichiers.

-Perspectives d'évolution :

Dans le futur nous comptons remédier aux limites de ce système en y ajoutant de nombreuses fonctionnalités, en transformant également ce système en application utilisable par tous ceux qui le désire.Se système a été conçu de sorte à ce que toute personne le désirant puisse l'utiliser et l'améliorer.

3. références

-[python - ValueError: invalid literal for int\(\) with base 10: " - Stack Overflow](#)

-[Corriger l'erreur Bytes-Like Object Is Required Not STR en Python | Delft Stack](#)

-[Comment configurer un serveur de test local ? - Apprendre le développement web | MDN](#)

-YouTube : Multithreaded-File-Transfer-using-TCP-Socket-in-Python

