Mr. Emmanuel NGUIMBUS

NGUH PRINCE

NDOCK CAREL

MOLO ANNE

CAHIER DE CHARGES POUR UN GESTIONNAIRE DE TACHES AVEC UNE INTERFACE GRAPHIQUE

SOMMAIRE

[1. Objectifs : 1](#_Toc164439039)

[2. Description du Projet : 1](#_Toc164439040)

[3. Exigences Fonctionnelles 2](#_Toc164439041)

[4. Exigences Techniques 3](#_Toc164439042)

[5. Exigences de performances 4](#_Toc164439043)

[6. Contraintes de délais 4](#_Toc164439044)

[a. Acteurs du projet 4](#_Toc164439045)

[b. Ressources du projet 5](#_Toc164439046)

[c. Planning du projet 7](#_Toc164439047)

[7. Critères d’acceptation 7](#_Toc164439048)

[8. Documentation et livrables 8](#_Toc164439049)

[9. Méthodologie de développement 8](#_Toc164439050)

# Objectifs :

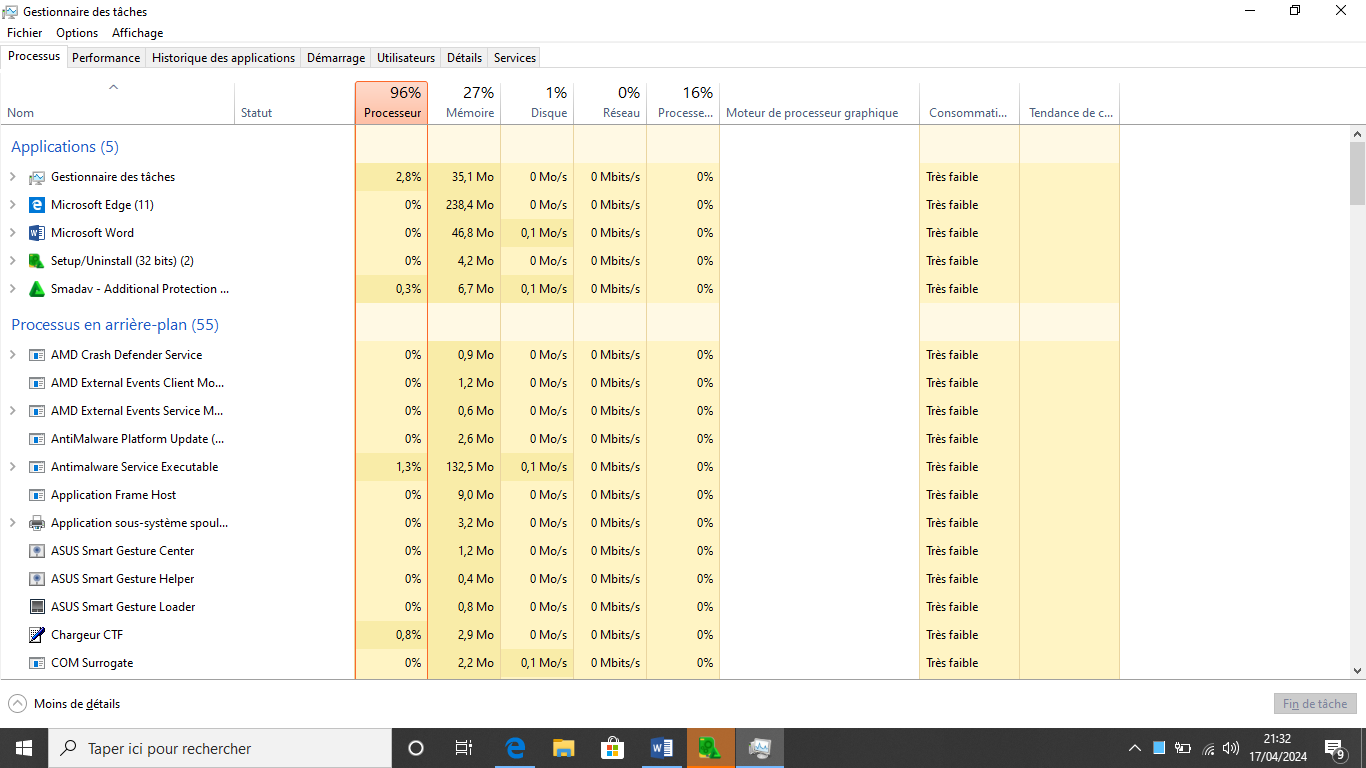
* Créer un Gestionnaire de tache avec une interface graphique.
* Empaqueter ce logiciel comme un paquet .rpm

# Description du Projet :

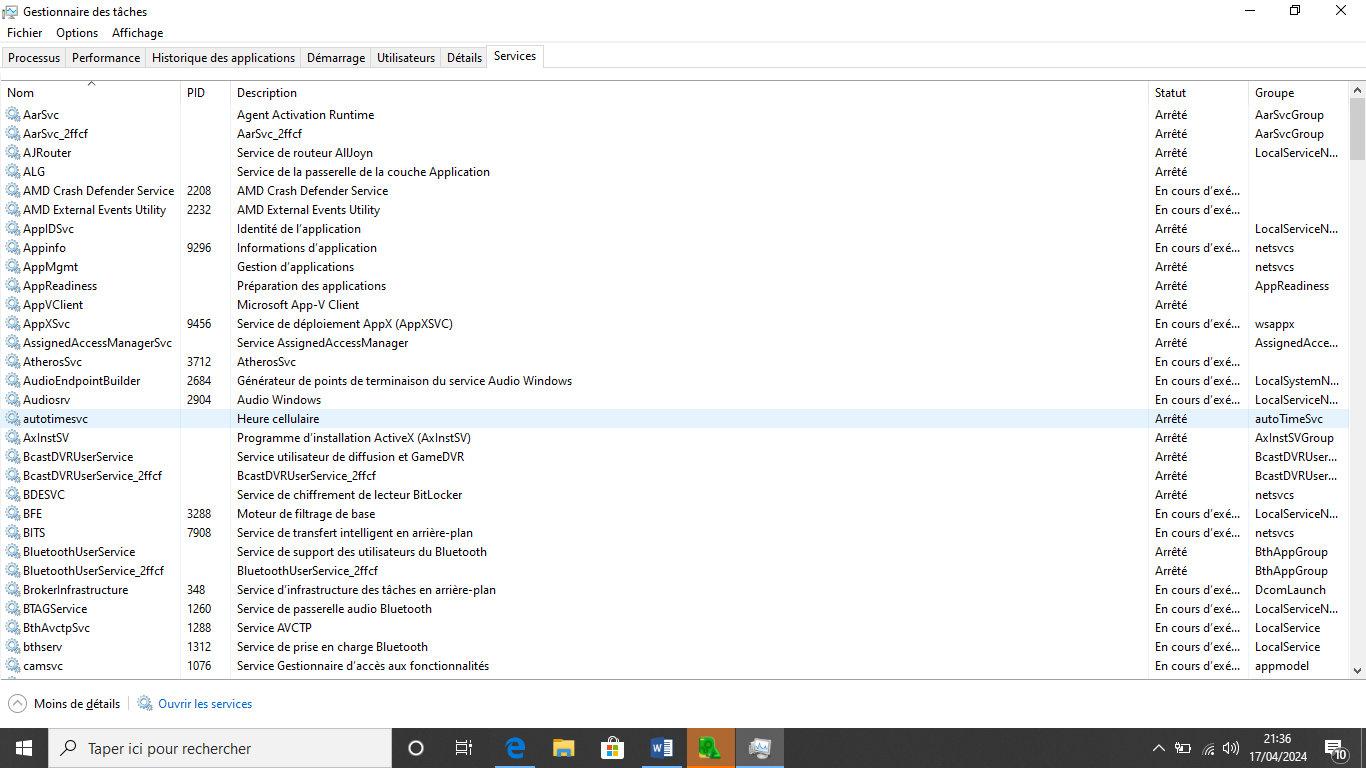
Cette application va tourner sur les systèmes d’exploitation Linux RPM-based (elle sera testée sur centOS) en tant que service qui fournit une interface graphique.

Le projet est un gestionnaire de tache offrant une interface graphique. Ce gestionnaire de tache va nous permettre de

* Voir tous les processus en cours d’exécution avec les ressources allouées



* Arrêter les processus
* Voir, arrêter, démarrer et redémarrer un service



# Exigences Fonctionnelles

* L’Application peur être accédé depuis l’invite de commande, cette application va nous permettre de :
  + Lister les processus en cours d’exécution
  + Arrêter les processus
  + Lister les services avec le PID
  + Arrêter, Démarrer et Redémarrer les services
* L’interface graphique doit être accessible en utilisant un raccourci clavier. Cette interface va nous permettre de faire les mêmes opérations que sur l’invite de commande
* L’interface graphique va nous afficher l’historique de processus (temps processeur et réseau)

# Exigences Techniques

* Empaqueter l’Application comme un paquet .rpm qui sera tester sur un système CentOS.
* Le paquet doit être téléchargeable depuis un dépôt GitHub
* Le projet aura deux Applications principales ; dont une partie Service et une partie Interface Utilisateur
* La partie service contient toutes les fonctionnalités attendues (gestion des processus, services, et historiques).
* La partie service sera écrit en Python et elle va fonctionner comme un serveur.
* La partie interface utilisateur sera écrit en Python. Cette partie ci dépend de la partie service (elle va fonctionner comme un client)
* L’application (la partie service précisément) ne doit pas être accessible hors de la machine. Celle-ci est une mesure de sécurité.
* L’application doit garder un historique des actions fait sur elle (par exemple redémarrage des services, arrêt des processus, etc.)

# Exigences de performances

* L’application doit répondre aux sollicitations des utilisateurs rapidement.
* L’application doit être accessible des le lancement du système d’exploitation.
* L’interface graphique doit présenter les données sur les processus et services a jour (par exemple les ressources consommes, les enfants des processus, les états des services etc.)
* La communication entre l’interface graphique est la partie service doit être instant et de tel sorte que la partie service peut envoyer des informations a l’interface graphique sans une requête venant de ce dernier.

# Contraintes de délais

## Acteurs du projet

|  |  |
| --- | --- |
| **Acteur** | **Rôle** |
| Mr. Emmanuel NGUIMBUS | Maitre d’ouvrage |
| MOLO ANNE | Chef d’analyse |
| NDOCK CAREL | Chef de conception |
| NGUH PRINCE | Chef développeur |

## Ressources du projet

* + 1. **Ressources humaines**

|  |  |  |
| --- | --- | --- |
| **Rôle** | **Durée** | **Cout** |
| Analyste | 14 | 15000 x 14 = 210000 |
| Concepteur | 14 | 15000 x 14 = 210000 |
| Développeur | 35 | 10000 x 35 = 350000 |
| Testeur | 15 | 5000 x 15 = 75000 |
|  |  |  |
| **Total** | | **845000** |

* + 1. **Ressources logicielles**

|  |  |  |
| --- | --- | --- |
| **Logiciel** | **Fonction** | **Cout (FCFA)** |
| Microsoft Word | Rédaction des documents | 50000 |
| Visual Studio Code | Editeur de texte, pour coder l’application | Gratuit |
| CentOS | Environnement de test | Gratuit |
|  |  |  |
|  |  |  |
| **Total** | | **50000** |

* + 1. **Ressources matérielles**

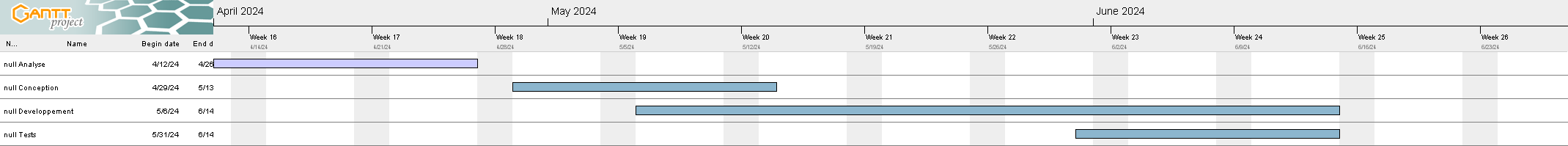
|  |  |  |
| --- | --- | --- |
| **Matériel** | **Fonction** | **Cout** |
| Machine desktop, core i7, 8gb de RAM DDR4, 500 GB SSD | Environnement de test | 250000 |
| Modem portable | Connexion internet, tests en réseau | 25000 |
| Connexion Internet (Camtel) | Recherche, publication de code, travail collaborative | 75000 |
|  |  |  |
| **Total** | | **350000** |

* + 1. **Total budget du projet**

**1,245,000**

## Planning du projet

* Le projet aboutira le 15 Juin 2024
* L’état de projet sera présenté chaque semaine jusqu’au délais final (les sprints). Apres cette présentation, l’équipe de développement va fixer les objectifs pour la semaine suivante.



# Critères d’acceptation

Cette partie du cahier de charges va lister les critères d’acceptation pour chaque partie de l’application

* **La partie service** 
  + Est capable de faire la gestion des processus (Lister et arrêter) à partir de l’invite de commande
  + Gestion des services (lister, arrêter, démarrer et redémarrer à partir de l’invite de commande
* **La partie interface utilisateur**
  + Gestion des processus et services à partie d’une interface graphique
* **Empaquetage** 
  + Tout le projet sera empaqueté comme un paquet .rpm (partie service et interface utilisateur)
  + En installant ce paquet, la partie service est configuré de démarrer automatiquement.

# Documentation et livrables

* Cahier de charge
* Cahier d’analyse
* Cahier de conception
* Guide d’utilisation
* Application empaqueté (.rpm)

# Méthodologie de développement

* **Méthodologie Scrum**