UVSQ université PARIS-SACLAY

IUT de Vélizy-Rambouillet
CAMPUS DE VÉLIZY-VILLACOUBLAY
CAMPUS DE RAMBOUILLET

Cahier des charges

V 02.00

Guenfici Rayane | Chedozeau Mathieu | Renouf Ugo | Belaidi Elyas | Bullock Patrick

0 - Table des matières :

1 - Introduction

Ce document représente le cahier des charges pour le développement de l'application web **LEARN TO ME !**, dédiée à la réalisation de diverses simulations dans différents domaines.

Divisé en cinq parties, dont cette introduction, il expose le problème à résoudre, les objectifs du projet, les livrables envisagés, les connaissances, ressources nécessaires, ainsi que les priorités de développement. Une section spécifique détaille également l'intégration des recommandations spécifiques issues du TP Python.

Ce document sera sujet à des évolutions durant le développement, et un système de versionnage sera utilisé pour faciliter le suivi chronologique.

Exemple :
- V XX.00 : Version "final" correspondant aux rendus à chaque sprint / livrables.
- V 00.XX : Corrections ou ajouts après publication, remise, ou versionnement du document.

2 - Énoncé

Le projet vise à mettre en place une plateforme permettant l'exécution de calculs distribués ou parallèles à partir de données prédéfinies ou transmises par l'utilisateur. Les résultats seront présentés sous forme de graphiques et/ou de données. De plus, l'application permettra à l'utilisateur de générer un modèle optimisé pour la prédiction de données en fonction des données transmises.

3 - Pré-requis

Les prérequis se décomposent en deux catégories :

Ressources Matérielles :

- Utilisation d'un "kit cluster hat" comprenant 4 Pi Zero et un Kit Cluster Hat.

Ressources Logicielles :

- Utilisation de GitLab pour le suivi et le versionnage du projet.
- Figma et la suite Adobe pour la conception graphique.
- Suite de logiciels JetBrains pour le développement.

4 - Intégration des Recommandations TP Python

Le projet intègre les recommandations spécifiques du TP Python que M.Hoguin nous a donné pour optimiser l'utilisation de la grappe de Raspberry Pi.

4.1 Configuration des Nœuds

- Renommer les nœuds comme suit :
  - Node1 (noeud principal)
  - Node2 (noeud secondaire)
  - Node3 (noeud secondaire)
  - Et ainsi de suite...

4.2 Clés SSH

- Établir une communication sécurisée :
- Générer une paire de clés sur chaque nœud ouvrier. Commande à utiliser : `ssh-keygen -t rsa`.
- Copier les clés au nœud maître. Commande à utiliser : `ssh-copy-id @IP-noeudmaître`.
- Répéter le processus sur le nœud maître et copier les clés sur tous les nœuds ouvriers.

4.3 Installation MPI

- Installer MPI sur chaque nœud. Commande à utiliser : `apt install mpich python3-mpi4py`.
- Vérifier le bon fonctionnement de MPI. Commande à utiliser : `mpiexec -n 1 hostname`.

4.4 Test en Grappe

- Tester la communication avec tous les nœuds depuis le nœud maître.
Commande à utiliser : `mpiexec -n 4 --hosts @ip1,@ip2,@ip3,@ip4 hostname`.

4.5 Intégration du Programme Python

- Enregistrer le code Python sur le nœud principal.
- Intégrer le programme dans l'application web pour l'exécution distribuée du calcul.

4.6 Propagation du Programme

- Copier le programme sur chaque nœud. Commande à utiliser : `scp prime.py @ip1`, `scp prime.py @ip2`, etc.
- Chaque nœud peut exécuter le programme en standalone pour vérifier son bon fonctionnement.

4.7 Distribution du Calcul aux RPI

- Lancer la commande pour répartir le calcul entre chaque RPI.
Commande à utiliser : `mpiexec -n 4 --host @ip1,@ip2,@ip3,@ip4 python3 prime.py 100000`.
Une fois terminé, les RPI renvoient les résultats au RPI maître.

5 - Priorités

La première phase du projet se concentrera sur la configuration des cartes SD pour assurer le bon fonctionnement du système et du "kit cluster hat". Ensuite, la mise en place de la plateforme web et le développement des programmes nécessaires aux calculs seront abordés dans un second temps.