



### **Cahier des charges SAE S5**

**Sommaire**

I.	Introduction	2
II.	Énoncé	2
III.	Pré-requis	2
IV.	Priorités	3
V.	Conclusion	3
VI.	Annexe	4

## I. Introduction

Ce projet a pour objectif de développer une plateforme web connectée à un cluster de Raspberry Pi, permettant d'exécuter des programmes de calcul distribué ou parallèle.

Inscrit dans un cadre pédagogique, ce projet vise à appliquer des compétences en programmation distribuée, administration de systèmes (Raspberry Pi en cluster), développement web et gestion d'infrastructures légères.

L'application web servira d'interface entre les utilisateurs et le cluster, afin de lancer des programmes, récupérer les résultats et les afficher de manière ergonomique. Elle devra également gérer différents profils d'utilisateurs et garantir la sécurité.

## II. Énoncé

La plateforme proposera un accès différencié selon le type d'utilisateur :

- Visiteurs : accès limité à une page d'accueil.
- Utilisateurs inscrits : possibilité de se connecter, lancer un programme de calcul distribué, suivre l'avancement de l'exécution et consulter les résultats sous forme de tableau ou de graphique. Ils disposent d'un espace personnel avec historique des exécutions et pourront gérer leur mot de passe.
- Administrateurs web : gestion des utilisateurs (création/suppression de comptes, import CSV), consultation des historiques, ajout ou suppression de programmes disponibles dans la bibliothèque.
- Administrateurs système : supervision technique du cluster (logs système, état des Raspberry, suivi des ressources utilisées, maintenance du cluster).

Les fonctionnalités principales incluent :

- Une interface web ergonomique pour le lancement et le suivi des calculs.
- Une sécurisation des accès (mots de passe chiffrés, CAPTCHA lors de l'inscription).
- Un suivi des activités via des logs (connexion, lancement de programme, déconnexion, suppression de compte).
- Une intégration transparente avec le cluster Raspberry, via SSH ou une API dédiée, pour répartir les tâches de calcul sur plusieurs nœuds.

### III. Pré-requis

La réalisation de ce projet nécessitera :

- Technologies de développement web : HTML, CSS, JavaScript (front-end), PHP ou Python Flask/Django (back-end).
- Gestion de bases de données : MySQL/MariaDB ou équivalent, pour stocker comptes, historiques et logs.
- Infrastructure matérielle : un cluster de Raspberry Pi (5 nœuds), chacun équipé d'une carte micro-SD et relié en réseau local, composé d'un Raspberry Pi 4 et de 4 Raspberry Pi 0.
  - Raspberry Pi 4 :
    - Architecture : aarch64
    - Nom de modèle : Cortex-A72
    - CPU(s) : 4
    - CPU min MHz : 600.0000
    - CPU max MHz : 1800.0000
  - Raspberry Pi 0 :
    - Architecture : armv6l
    - Nom de modèle : ARM1176
    - CPU(s) : 1
    - CPU min MHz : 700.0000
    - CPU max MHz : 1000.0000
- Logiciels nécessaires : serveur web (Apache ou Nginx), Python (bibliothèques de calcul distribué comme mpi4py ou équivalent), Java (Socket TCP).
- Compétences attendues :
  - Développement web full-stack.
  - Administration Linux (sécurisation SSH, gestion réseau, monitoring).
  - Programmation parallèle/distribuée (Python + MPI, Java Socket).
  - Gestion et analyse de logs.

### IV. Priorités

Mise en place du cluster Raspberry : configuration réseau, installation des outils Python et Java pour le calcul distribué, vérification du bon fonctionnement du cluster.

Développement des programmes de calcul fournis et tests de leur exécution en mode distribué.

Création de l'interface web de base : authentification, tableau de bord utilisateur, lancement d'un programme.

Ajout des fonctionnalités avancées :

- Historique des résultats.
- Gestion utilisateurs (par l'administrateur web).
- Monitoring du cluster (pour l'administrateur système).

Sécurisation de l'ensemble : chiffrement des mots de passe, CAPTCHA, traçabilité via logs.

Optimisation et ergonomie : présentation claire des résultats (graphiques, tableaux), amélioration UI/UX.

## **V. Conclusion**

En conclusion, ce projet vise à proposer une plateforme de calcul distribué pédagogique exploitant un cluster Raspberry Pi.

Il permettra de mettre en pratique des compétences techniques variées allant du développement web à la programmation parallèle et à l'administration système.

En suivant les étapes de priorisation, le projet garantira la mise en place d'une solution sécurisée, fonctionnelle et évolutive, adaptée aux besoins des différents utilisateurs.

**VI. Annexe**Lecture du cahier des charges

Objets	Acteurs	Actions
Interface Web	Utilisateur	Se connecter/S'inscrire sur la plateforme
Interface Web	Utilisateur	Déposer un programme Python à exécuter
Interface Web	Utilisateur	Lancer l'exécution sur le cluster
Interface Web	Utilisateur	Consulter les résultats du calcul
Interface Web	Administrateur	Gérer les comptes utilisateurs
Cluster Raspberry Pi	Système	Répartir le calcul sur les nœuds du cluster
Cluster Raspberry Pi	Système	Exécuter les tâches reçues du serveur web
Cluster Raspberry Pi	Système	Transmettre les résultats au serveur principal
Serveur Web/API	Système	Centraliser les requêtes utilisateurs
Serveur Web/API	Système	Distribuer les calculs sur le cluster
Serveur Web/API	Système	Collecter les résultats et les renvoyer à l'interface
Base de données	Système	Enregistrer les utilisateurs, programmes et résultats
Base de données	Administrateur	Consulter les logs et statistiques d'utilisation
Nœud Raspberry Pi	Système	Recevoir la tâche et exécuter la partie du programme
Nœud Raspberry Pi	Système	Communiquer l'état et les résultats au serveur maître
Programme Python	Système	lancer le programme et gérer le calcul distribué

Questions pour lever les ambiguïtés

- 1) Quels types de calculs seront exécutés ?
- 2) Quelle serait la durée typique attendue ?
- 3) Doit-on faire un administrateur web et un administrateur système ou un seul administrateur ?
- 4) Souhaitez vous un historique des calculs par utilisateur et que celui-ci devrait contenir ?