



IUT de Vélizy-Rambouillet
CAMPUS DE VÉLIZY-VILLACOUBLAY

Recueil des exigences SAE S5

Sommaire

I) Objectif et portée	3
a) Quels sont la portée et les objectifs généraux ?	3
b) Les intervenants	3
c) Qu'est-ce qui entre dans cette portée ? Qu'est-ce qui est en dehors ?	3
II) Terminologie employée / Glossaire	5
III) Cas d'utilisations	7
IV) Technologie employée	10
V) Autres exigences	10
a) Processus de développement	10
i) Qui sont les participants au projet ?	10
ii) Quelles valeurs devront être privilégiées ?	10
iii) Quels retours ou quelle visibilité sur le projet les utilisateurs et commanditaires souhaitent-ils ?	10
iv) Que peut-on acheter ? Que doit-on construire ?	
Qui sont nos concurrents ?	11
v) Quelles sont les autres exigences du processus ?	11
vi) A quelle dépendance le projet est-il soumis ?	11
b) Règles métier	11
c) Performances	11
d) Opération, sécurité, documentation	11
e) Utilisation et utilisabilité	11
f) Maintenance et portabilité	11
g) Questions non résolues ou reportées à plus tard	11
VI) Recours humains, questions juridiques, politiques, organisationnelles	
a) Quel est le recours humain au fonctionnement du système ?	12
b) Quelles sont les exigences juridiques et politiques ?	12
c) Quelles sont les conséquences humaines de la réalisation du système ?	12
d) Quels sont les besoins en formation ?	12
e) Quelles sont les hypothèses et les dépendances affectant l'environnement humain ?	12

I) Objectif et portée

a) Quels sont la portée et les objectifs généraux ?

L'objectif de ce projet est de développer une application web permettant à un utilisateur inscrit de lancer des scripts de calculs parallèle au sein d'un cluster de Raspberry Pi, dans le cadre d'un projet d'IUT.

Objectifs généraux :

- Réaliser les pages web du site en HTML/CSS
- Développer l'application web avec PHP et MySQL pour la base de donnée utilisateur
- Gérer différents types d'utilisateurs (visiteurs, utilisateur inscrit, administrateur web, administrateur système)
- Installation et configuration du cluster de Raspberry Pi pour héberger l'application web
- Développement des différents programmes de calcul en Python et Java

b) Les intervenants

- Visiteur : le visiteur n'a accès qu'à la page d'accueil du site, il peut visionner la vidéo de présentation du site, il peut également créer ou se connecter à un compte
- Utilisateur inscrit : Suite à s'être connecté, les visiteurs ont accès à leurs profils, modifier leurs informations personnelles, aux modules calculs ainsi qu'à leur historique
- Administrateur web : Peut gérer les comptes des autres utilisateurs (créer, modifier, supprimer)
- Administrateur système : Peut faire des installations de nouveaux modules et modifier les fichiers du site

c) Qu'est-ce qui entre dans cette portée ? Qu'est-ce qui est en dehors ?

Portée du système :

1. Développement de l'application web :
 - Création d'une interface utilisateur accessible via un navigateur, permettant d'effectuer divers calculs sur des modules définis.
 - Utilisation de PHP et MySQL pour le back-end de l'application.
 - Utilisation de HTML et CSS pour le frontend du site, afin de rendre l'application esthétique et fonctionnelle.
2. Scripts de calculs partagés :
 - Création de différents scripts de calculs (prime.py, Monte Carlo, calcul intégral)
 - Chaque script sera accessible après connexion des utilisateurs et pourra être enrichi au fur et à mesure des besoins.
3. Gestion des utilisateurs :

- Système d'authentification permettant à un utilisateur de se connecter, modifier ses informations personnelles et accéder à ses résultats.

- Création d'un système d'administration pour gérer les utilisateurs, les rôles et les permissions.

4. Interface administrateur :

- Développement d'un tableau de bord pour les administrateurs web afin qu'ils puissent gérer les comptes des utilisateurs (création, modification, suppression).

- Mise en place de permissions d'accès pour les différents types d'utilisateurs (visiteurs, utilisateurs inscrits, administrateurs).

5. Hébergement sur Raspberry Pi :

- Configuration du Raspberry Pi pour héberger l'application web.

- Installation de logiciels nécessaires comme un serveur web (Apache ou Nginx), PHP et MySQL.

- Configuration de l'environnement d'exécution pour que l'application soit accessible sur le réseau local ou via internet (si applicable).

6. Sécurité des données :

- Mise en place de mécanismes de sécurité comme le chiffrement des mots de passe, la validation des entrées pour éviter les attaques (SQL injection).

Limites du système :

1. Applications mobiles :

- L'application est conçue pour une utilisation via un navigateur web. La création d'applications natives pour iOS ou Android n'est pas incluse dans ce projet.

2. Gestion de la haute disponibilité et du cloud :

- La gestion de la haute disponibilité (scalabilité horizontale) ou le déploiement sur des plateformes cloud comme AWS, Azure ou Google Cloud ne fait pas partie de la portée initiale, surtout avec l'hébergement sur un Raspberry Pi qui est limité en termes de ressources.

3. Maintenance à long terme :

- Le projet ne couvre pas la maintenance à long terme ou les mises à jour régulières une fois que la plateforme est mise en production.

- Aucun contrat de support technique ou de gestion des incidents n'est inclus.

4. Personnalisation par l'utilisateur final :

- Les utilisateurs ne pourront pas créer leurs propres modules de calcul ni personnaliser les fonctionnalités au-delà de leur profil personnel et de leurs résultats de calcul.

5. Support multilingue :

- L'application ne sera probablement pas disponible dans plusieurs langues dès le départ, bien que cette fonctionnalité puisse être ajoutée dans le futur.

II) Terminologie employée / Glossaire

SAé (Situation d'Apprentissage et d'Évaluation) : Projet permettant aux étudiants de mettre en pratique des compétences techniques et théoriques acquises durant l'année.

RPI (Raspberry Pi) : Ordinateur monocarte utilisé pour héberger le serveur web du projet et exécuter l'application développée.

Github : Plateforme de gestion de versions utilisée pour héberger le code source du projet et permettre le suivi des modifications.

Administrateur web : Personne responsable de la gestion des utilisateurs inscrits, des modules de calculs et de la maintenance des comptes via l'interface d'administration.

Administrateur système : Personne en charge de la gestion du serveur, notamment de la configuration, du déploiement, et de la maintenance des services associés à la plateforme web

Python : Langage de programmation interprété, multiparadigme et multiplateformes. Il est conçu pour optimiser la productivité des programmeurs en offrant des outils de haut niveau et une syntaxe simple à utiliser.

MPI (Message Passing Interface) : Outil pour le calcul scientifique à haute performance qui permet d'utiliser plusieurs ordinateurs. C'est une norme conçue en 1993-94 pour le passage de messages entre ordinateurs distants ou dans un ordinateur multiprocesseur. Elle est devenue un standard de communication de facto pour des nœuds exécutant des programmes parallèles sur des systèmes à mémoire distribuée.

Java : Langage de programmation de haut niveau orienté objet. Un logiciel écrit en langage Java a pour particularité d'être compilé vers un code intermédiaire formé de bytecodes qui peut être exécuté dans une machine virtuelle Java (JVM) en faisant abstraction du système d'exploitation.

Socket : Élément logiciel qui est aujourd'hui répandu dans la plupart des systèmes d'exploitation. Il s'agit d'une interface logicielle avec les services du système d'exploitation, grâce à laquelle un développeur exploitera facilement et de manière uniforme les services d'un protocole réseau. Il lui sera ainsi par exemple aisé d'établir une session TCP, puis de recevoir et d'expédier des données grâce à elle.

TCP (Transmission Control Protocol) : Protocole de transport fiable, en mode connecté fonctionnant en trois phases : l'établissement de la connexion, les transferts de données et la fin de la connexion.

SSH (Secure Shell) : Protocole de communication sécurisé. Le protocole de connexion impose un échange de clés de chiffrement en début de connexion. Par la suite, tous les segments TCP sont authentifiés et chiffrés.

CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart) : Famille de tests de Turing permettant de différencier de manière automatisée un

utilisateur humain d'un ordinateur. Ce test d'authentification de type défi-réponse est utilisé en informatique pour vérifier que l'utilisateur n'est pas un robot

HTML (HyperText Markup Language) : Langage de balisage conçu pour écrire les pages web.

CSS (Cascading Style Sheets) : Langage informatique qui décrit la présentation des documents HTML et XML.

PHP (Hypertext Preprocessor) : PHP est un langage de script utilisé le plus souvent côté serveur : dans cette architecture, le serveur interprète le code PHP des pages web demandées et génère du code et des données pouvant être interprétés et rendus par un navigateur web.

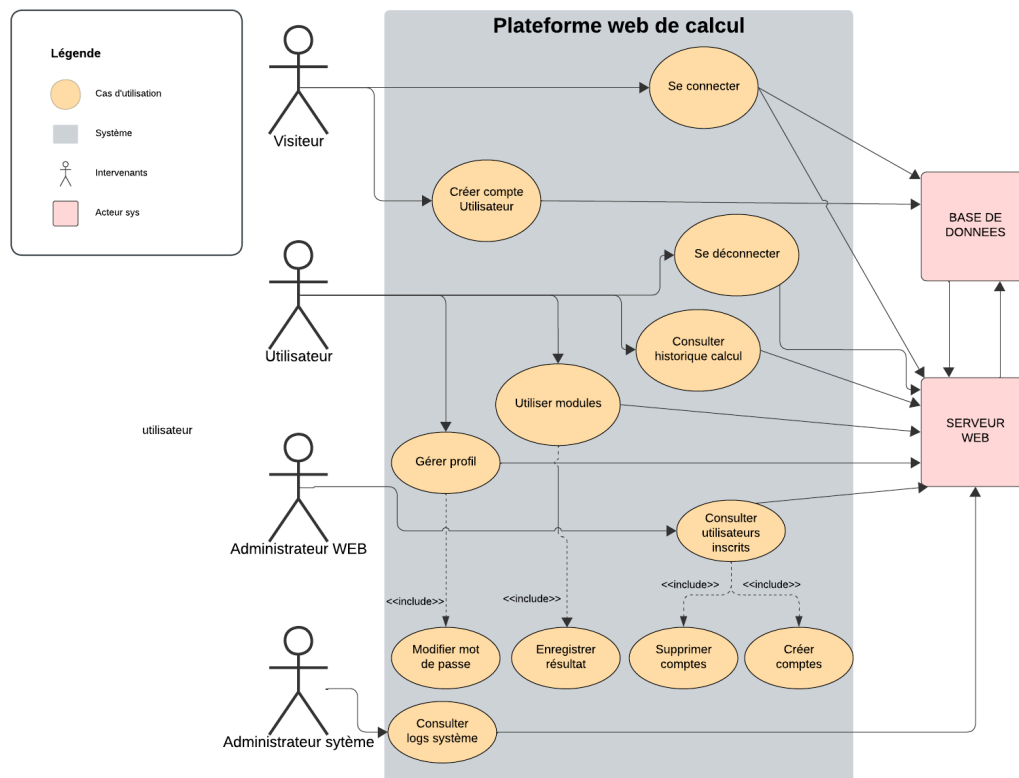
MySQL : système de gestion de bases de données relationnelles (SGBDR).

Programmation partagée : modèle de programmation où plusieurs processus ou threads accèdent à une même zone de mémoire.

Programmation distribuée : concerne des programmes dont les composants s'exécutent sur plusieurs machines distinctes, reliées par un réseau.

III) Cas d'utilisations

Le diagramme comprenant les acteurs principaux et leurs objectifs :



Lien :

https://lucid.app/lucidchart/0396fa81-ce6c-4a1f-97a1-1d0d16478c6c/edit?invitationId=inv_54497026-5345-4f18-9af9-9367bdc4bf39&page=.Q4MUjXso07N#

Exemple de cas d'utilisations :

Cas d'utilisation :	Créer compte utilisateur
Description	Un visiteur veut créer un compte utilisateur pour accéder à l'interface permettant de faire des calculs
Portée	Système
Niveau	Utilisateur
Acteur Principale	Visiteur
Scénario nominal	<ol style="list-style-type: none"> 1. Le visiteur se rends sur le formulaire d'inscription 2. Le visiteur rentre ses informations 3. Une confirmation est affiché au visiteur
Scénario alternatifs	
Scénario exceptionnel	<ol style="list-style-type: none"> 1. Le login existe déjà <ol style="list-style-type: none"> a. Le visiteur se rends sur le formulaire d'inscription b. Le visiteur rentre ses informations c. Le visiteur valide le captcha d. Renvoie une erreur lui indiquant que le login est déjà pris 2. Le login ne possède pas le nombre de caractères requis <ol style="list-style-type: none"> a. Le visiteur se rends sur le formulaire d'inscription b. Le visiteur rentre ses informations c. Le visiteur valide le captcha d. Renvoie une erreur lui indiquant que le login ne possède pas le nombre nécessaire de caractères
Pré-condition	
Post-condition	Un compte utilisateur a été créé.

Cas d'utilisation :	Se connecter
Description	Un visiteur veut créer un compte utilisateur pour accéder à l'interface permettant de faire des calculs
Portée	Système
Niveau	Utilisateur
Acteur Principale	Visiteur
Scénario nominal	<ol style="list-style-type: none"> 1. Le visiteur se rends sur le formulaire d'inscription 2. Le visiteur rentre ses informations 3. (Le visiteur valide le captcha) Optionnel 4. Une confirmation est affiché au visiteur
Scénario alternatifs	
Scénario exceptionnel	<ol style="list-style-type: none"> 1. Le login existe déjà <ol style="list-style-type: none"> a. Le visiteur se rends sur le formulaire d'inscription b. Le visiteur rentre ses informations c. Le visiteur valide le captcha d. Renvoie une erreur lui indiquant que le login est déjà pris 2. Le login ne possède pas le nombre de caractères requis <ol style="list-style-type: none"> a. Le visiteur se rends sur le formulaire d'inscription b. Le visiteur rentre ses informations c. Le visiteur valide le captcha d. Renvoie une erreur lui indiquant que le login ne possède pas le nombre nécessaire de caractères
Pré-condition	
Post-condition	Un compte utilisateur a été créé.

Voir plus de cas d'utilisation dans l'annexe recueil_des_besoins.md.

IV) Technologie employée

L'application web utilise un cluster de Raspberry Pi composé d'un Raspberry Pi 4 et de quatre Raspberry Pi 0 pour effectuer du calcul parallèle.

- 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

V) Autres exigences

a) Processus de développement

i) Qui sont les participants au projet ?

- Florian Bouchery
- Esteban Colombani
- Luca Da Silva
- Noah Lavocat
- Roméo Leblond

ii) Quelles valeurs devront être privilégiées ?

- Performance
- Rapidité
- Fiabilité

iii) Quels retours ou quelle visibilité sur le projet les utilisateurs et commanditaires souhaitent-ils ?

Les utilisateurs souhaitent pouvoir utiliser des scripts de calcul qui s'effectueront en parallèle pour qu'ils prennent le moins de temps possible. Les commanditaires souhaitent pouvoir tester les résultats de nos calculs ainsi que la vitesse d'exécution. Ils souhaitent également un rapport et une documentation utilisateur.

iv) Que peut-on acheter ? Que doit-on construire ? Qui sont nos concurrents ?

A acheter : Rien à acheter

A construire :

- Application web
- Infrastructure logiciel du cluster
- Programmes de calculs partagés

Concurrent : Aucun

v) Quelles sont les autres exigences du processus ?

- Utilisation d'un système de gestion de version
- Respect des bonnes pratiques de développement (tests unitaires, documentation)

vi) A quelle dépendance le projet est-il soumis ?

- Accès au cluster de Raspberry Pi
- Compétences techniques de l'équipe
- Calendrier universitaire

b) Règles métier

- Chaque tâche de calcul doit être découpée en sous-tâches et distribuée entre les Raspberry Pi.
- Les résultats doivent être rassemblés et restitués via l'application web.
- Les utilisateurs ne doivent pas exécuter de code non autorisé sur le cluster.

c) Performances

- Temps d'exécution des calculs distribué inférieur à un calcul séquentiel.
- Latence entre la soumission et l'exécution d'une tâche minimale.

d) Opération, sécurité, documentation

- Interface web protégée par authentification.
- Documentation utilisateur et technique (installation, utilisation, architecture).

e) Utilisation et utilisabilité

- Interface intuitive : boutons clairs, retours visuels sur les actions.
- Compatibilité avec les principaux navigateurs.
- Documentation d'aide intégrée ou tutoriel.

f) Maintenance et portabilité

Pas de maintenance prévue pour le moment

g) Questions non résolues ou reportées à plus tard

Aucune

VI) Recours humains, questions juridiques, politiques, organisationnelles

a) Quel est le recours humain au fonctionnement du système ?

Développeurs / équipe projet : assurent la conception, le développement et les tests de l'application web et du module de calcul distribué.

Utilisateurs finaux : utilisent l'application web pour soumettre des calculs, consulter les résultats.

b) Quelles sont les exigences juridiques et politiques ?

- Respect des licences logicielles :

Tous les composants utilisés doivent être conformes à leurs licences respectives.
Aucun logiciel à licence commerciale ne doit être intégré sans autorisation.

- Protection des données :

L'application web doit garantir la confidentialité des données saisies.

c) Quelles sont les conséquences humaines de la réalisation du système ?

Le projet constitue une expérience d'apprentissage concrète, renforçant les compétences des étudiants en systèmes distribués, développement web, et travail collaboratif.

d) Quels sont les besoins en formation ?

- Installation et configuration d'un Raspberry Pi (OS, SSH, réseau).
- Introduction au calcul distribué (concepts de master / workers, tâches parallèles).
- Utilisation de Git pour la gestion collaborative du code.
- Déploiement et administration de l'application web (framework, serveur).
- Guide rapide d'utilisation de l'application web
- Documentation simplifiée

e) Quelles sont les hypothèses et les dépendances affectant l'environnement humain ?

- Disponibilité et implication de l'équipe de développement
- Accès au matériel
- Compétences techniques des utilisateurs