Application de sélection de repas

Cahier des charges

Équipe **01** :
Ackermann Simon (Développeur Backend simon.ackermann@heig-vd.ch)
Boegli Noah (Développeur Backend, remplaçant chef de projet noah.boegli@heig-vd.ch)
Comte Emmanuelle (Tests techniques et gestion des données emmanuelle.comte@heig-vd.ch)
Gachet Jean (Développeur Backend jean.gachet@heig-vd.ch)
Gallay David (Chef de projet/Product Owner, review fonctionnelle/qualité david.gallay@heig-vd.ch)
Mendes Hugo (Développeur Frontend hugo.mendes@heig-vd.ch)
Terrani Fabien (Développeur Frontend et responsable UI/UX fabien.terrani@heig-vd.ch)
(ordre alphabétique)

Présentation du projet

Contexte

Expliquez ici les raisons de l'existence de ce projet.
Environnement dans lequel s'inscrit le projet (stratégie, enjeux, domaine, etc.)
Quel est le problème à résoudre ? Qui intéresse-t-il et pourquoi?

"Qu'est-ce que l'on mange?". C'est une question que l'on se pose souvent, que l'on mange seul ou à plusieurs.

- On choisit par simplicité, sans pour autant être satisfait
- On mange souvent la même chose, pas équilibré.
- On peine à se mettre d'accord lors de repas en groupe. (Envies, allergies, éthiques, …)
- On achète des produits "couteaux-suisses", comme des pâtes ou du surgelé qui durent longtemps, se cuisinent facilement. Cela a notamment un impact sur le budget et le gaspillage.

Ce sont des problèmes qui concernent tout le monde. On pense principalement aux personnes ayant la responsabilité du repas, mais peu de temps à y accorder comme la tranche d'âge 18-35 ans, dont les étudiants, mais on peut à tout âge vouloir organiser un repas à plusieurs.

Le but de ce projet est de permettre aux utilisateurs de choisir des repas, contrairement aux applications classiques qui proposent des recettes une fois que le repas est choisi: pour soi, pour un groupe, pour tout de suite comme pour plus tard. Les conséquences sont:

- Des repas meilleurs et plus variés pour une plus grande satisfaction.
- Un budget mieux contrôlé
- Moins de temps perdu sur les décisions, c'est du temps gagné pour autre chose.

Objectifs

Résultats que le projet doit atteindre
Distinguer les objectifs "must have" et les objectifs "nice to have" (si le temps le permet).
Un objectif répond à une attente d'une population cible (utilisateur et/ou exploitant)

L'application doit permettre à l'utilisateur de planifier des repas seuls ou en groupe de manière efficace. Pour cela, l'utilisateur définit son profil et peut créer/rejoindre un groupe. Au sein d'un groupe, les utilisateurs ont la possibilité de proposer des repas et de voter. Une personne peut aussi simplement choisir pour les autres en ayant un filtrage qui tient compte du profile de chaque utilisateur. Pour une utilisation individuelle, elle permet aux utilisateurs d'ajouter leurs propres recettes, de voter pour celle qu'ils préfèrent, avertir qu'une recette contient une erreur, etc.

Périmètre

Fixer les limites : qu'est-ce qui n'est pas un objectif ? qu'est-ce qui sort des limites, par itération)
Ce projet n'est pas un concurrent aux réseaux sociaux actuels tels que Facebook, Instagram, Snapchat, Linkedin. Il ne s'agit pas non plus d'un e-commerce ou d'un service de livraison de nourriture, bien que rien n'empêcherait une fonctionnalité d'accéder à ce genre de service à travers l'application. Elle permet principalement de chercher/se mettre d'accord sur un plat et une recette.

Description de l'existant

Composants logiciels et matériels sur lesquels se base le logiciel à développer : langage de programmation, bibliothèques, matériel particulier, ...
Système existant s'il s'agit d'une extension d'un système

Il s'agit d'une application web "LAMP" découpée en backend/frontend:

- Apache2-Php avec le framework Symfony pour le backend.
- Javascript avec Angular pour le frontend
- MariaDB comme base de données.
- OS Linux

Quelques remarques:

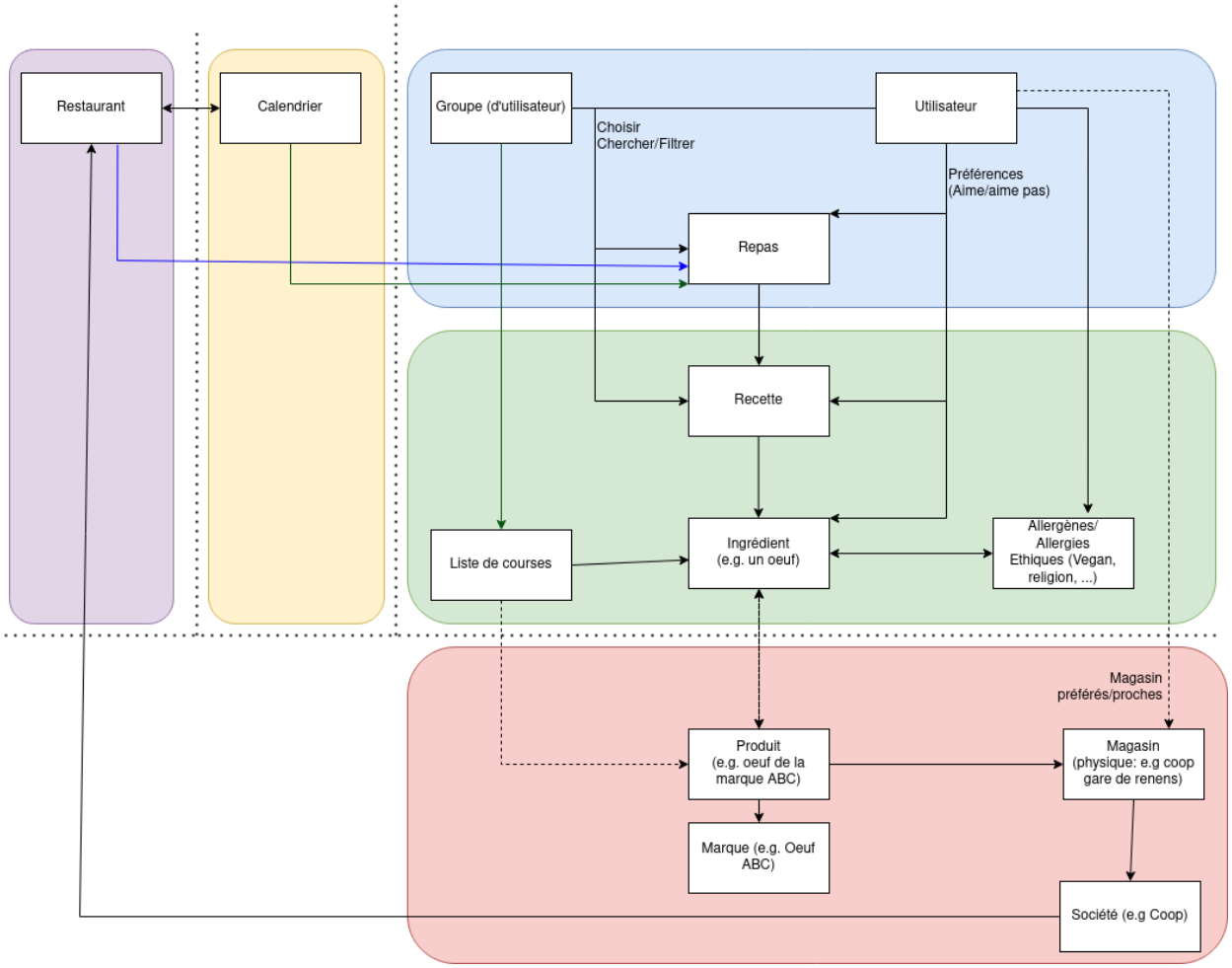
- Les choix ont été faits selon les compétences et envies des membres de l'équipe afin d'être le plus efficaces, toujours dans la mesure où la technologie est admise comme convenable pour le projet.
- Le choix du SGBD portait sur MariaDB et Postgresql. Le choix du SQL était assez claire vu les besoins fonctionnels. Il a penché en faveur de MariaDB pour son ancienneté et son excellente intégration avec php, mais aussi en tenant compte des offres de marché fournissant du DBaaS.
- L'application est actuellement très monolithique dans sa structure. Nous avons donc mis de côté docker qui ne serait pas d'un grand gain, ni pour le développement, ni pour la production. Nous nous tournons plutôt vers l'outil vagrant pour le développement qui correspond mieux à une petite équipe avec différents OS. Le déploiement de production sera à la charge d'une personne qui aura la liberté d'utiliser la méthode qui convient le mieux (manuellement, script, ansible, …)

Expression des besoins

Besoins fonctionnels

Fonctions (ou opérations, ou encore transformations) que le logiciel doit réaliser.
Les spécifications fonctionnelles peuvent être classées par importance.

Ci-dessous une modélisation conceptuelle de haut niveau des possibilités envisagées pour le service sur le long terme.



Dans le cadre de la période de temps accordé, seuls les encadrés bleu et vert seront traités.
Les fonctionnalités suivantes seront implémentées:

1. User sign up & login
2. Gestion de son profil utilisateur
    a. Préférences

    b. Allergies

    Nb: D'autres critères pourront être ajoutés, comme un budget ou des calories par repas.
3. Gestion de groupe
    a. Création de groupe
    b. Invitation de membre
    c. Selection du repas
        i. Proposition
        ii. Vote
        iii. Sélection aléatoire
4. Gestion des recettes
    a. Les recettes peuvent être ajoutées par les utilisateurs
    b. Les recettes sont modérées et notées par les utilisateurs
5. Recherche avancée sur les repas et recettes
    a. Tenant compte des profiles utilisateurs
    b. Ajout de filtres supplémentaires
6. Gestion d'une liste de course partagée au sein d'un groupe

Besoins non fonctionnels

Les spécifications non fonctionnelles sont toutes les spécifications qui n'expriment pas une fonction du logiciel (performance, fiabilité, facilité d'utilisation, maintenabilité, système d'exploitation cible, ...).

- Il n'y a pas un gros besoin en termes de performance.
    - Les pics d'utilisateurs seront en principe aux heures de repas
    - Il n'y a pas de fonctionnalité prévue qui soit gourmande en ressource.
    - Il n'y a pas de fonctionnalité prévue qui pourrait être amenée à être largement sollicitée.
- L'application est prévu pour être centralisée et n'a besoin de pouvoir tourner que sur une seule plateforme. En revanche, dans la majorité des scénarios, les utilisateurs accèdent au service à travers leurs smartphones. C'est pourquoi l'UX est essentiel
- Le succès de ce genre d'applications repose sur sa facilité d'utilisation. L'UX doit donc être extrêmement bien pensée. La priorité d'une fonctionnalité sera revue si son utilisation est trop complexe ou ne convient pas.
- L'application répondra rapidement à énormément de besoins avec un set limité de fonctionnalités mais beaucoup de fonctionnalités peuvent être pensées et imaginées comme extensions à l'application. Il est important que l'application soit maintenable pour permettre ces ajouts.

Déroulement du projet

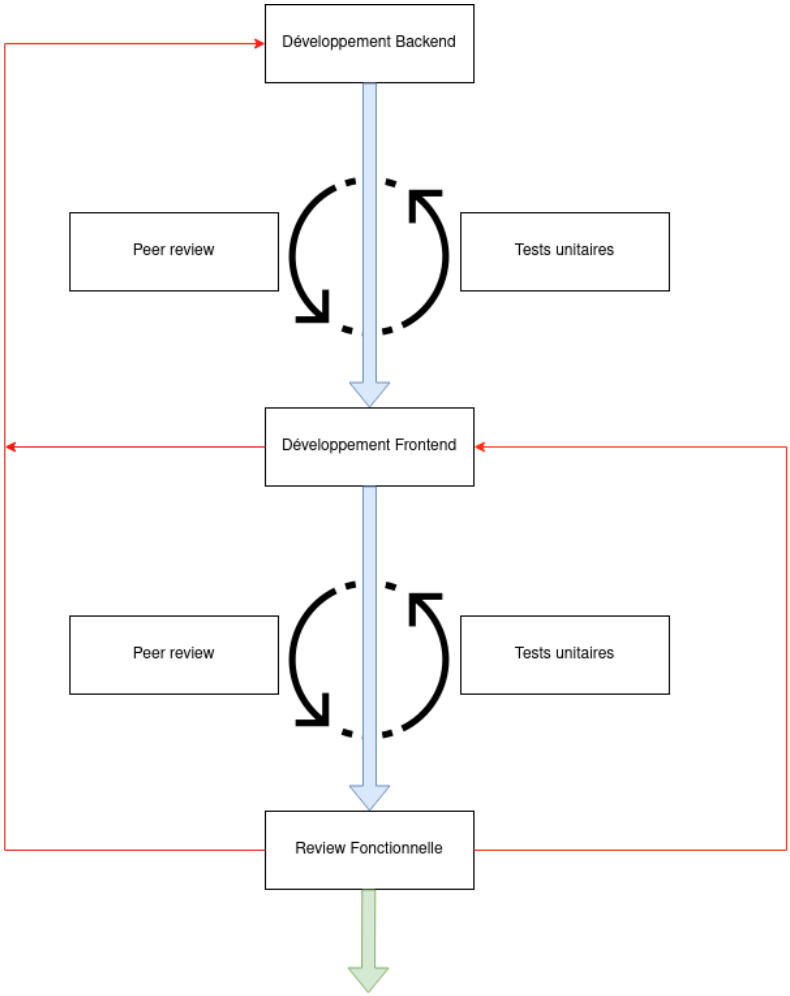
Planification

Articulation des grandes phases du projet et des principaux jalons
Estimation de la charge globale et répartition

Plan d'assurance qualité

Procédures adoptées pour contrôler la qualité des processus et livrables
La validation du produit sera faite à plusieurs niveaux:

- Le code est évalué en peer-review
- Les tests unitaires sont écrits par un tiers sur la base unique de la documentation du code.
- Chaque phase de développement (backend, frontend) doit valider la peer-review et les tests unitaires.
- La documentation fait partie des critères de validation d'une étape.
- Une fois l'aspect technique validé, une review fonctionnelle a lieu. La personne chargée n'a absolument pas connaissance du code et n'ont accès qu'à la documentation utilisateur et au changelog fonctionnel.



Documentation

Description de la documentation devant accompagner le logiciel à sa livraison
Différentes documentations pour différents publics: utilisateurs du logiciel, administrateurs du logiciel, développeurs du logiciel

Il y a plusieurs documentations qui seront rédigées par des personnes différentes:

- Les développeurs sont responsables de la documentation de leur code.
- La documentation du déploiement de développement et de production sera faite par la personne chargée du déploiement.
- Le chef de projet documente les fonctionnalités. La documentation résultante sera complétée par le responsable de l'UX. Ce document pourra servir aux end-user.