UNIVERSITÉ DE TECHNOLOGIE DE TROYES

MEALSQUARE

Encyclopédie de recette de cuisine

ARCHITECTURE LOGICIELLE

Étudiants

Méhéza SAMIE
André Marvell IKOUNGA

Responsable

Sophie Loriette

SOMMAIRE

Après avoir effectué l'analyse de ce projet, nous allons nous consacrer à l'étude de l'architecture logicielle. Dans cette partie, nous présenterons l'architecture fonctionnelle du futur service, de l'analyse de l'environnement logiciel à l'ensemble des outils de développement qui seront utilisés pour mettre sur pied ce projet.

Architecture Serveur

Présentation des outils

Comme nous l'avons précisé dans la phase d'analyse des besoins, le projet sera développé sous forme d'un site internet. Pour développer ce site, nous avons choisi le langage de programmation PHP en tant que principale technologie.

Nous justifions ce choix par le fait que le langage PHP est le langage de programmation Web le plus utilisé. Il existe une communauté de développeurs très active qui rend disponibles des dizaines de milliers de librairies PHP de grande qualité ainsi qu'une vaste quantité de documentations et tutoriels pour le bénéfice de chacun. Ces ressources faciliteront notre travail et réduiront notre temps d'exécution.

1. Un Framework PHP : Symfony2

Pour ne pas s'arrêter à la simple utilisation du langage PHP, nous avons décidé d'utiliser un framework PHP. Les frameworks confèrent plusieurs avantages parmi lesquels :

- ***Une organisation claire du projet :*** *L*es frameworks permettent d'améliorer considérablement l'organisation d'un projet, en permettant notamment :
  - ❏ Un découpage logique du code source
  - ❏ Une factorisation de composants communs, et une réutilisabilité du code
  - ❏ La maintenance et l'évolution faciles du projet

- ***Des composants et bibliothèques réutilisables*** : Les frameworks viennent systématiquement avec leur lot de composants réutilisables. Ces modules, plugins, ou bundles sont l'un des atouts majeurs de ces frameworks car ils permettent de gagner du temps sur le développement de certaines fonctionnalités.

- ***Une incitation aux bonnes pratiques de développement*** : Les frameworks PHP actuels ne cessent de mettre en avant ces bonnes pratiques afin de rendre compatible le code issu de différents frameworks. Ces bonnes pratiques permettent la réalisation de codes lisibles et compréhensibles pour tous les développeurs.

- ***Une base régulièrement mise à jour*** : derrière chaque framework, il y a une communauté active qui détecte et corrige quotidiennement des failles ou des manques du framework. Cela permet de profiter des mises à jour du framework ainsi que l'ensemble d'améliorations qu'elles comportent.

- Pour ne citer que ceux là…

C'est dans cette visée que notre choix s'est porté vers le framework PHP Symfony2, l'un des frameworks PHP les plus répandus sur le marché du développement web.

2. Un ORM : Doctrine

L'un des atouts majeurs du framework Symfony2 est l'utilisation d'un ORM (Object-Relational Mapping).

Un ORM est une classe (ou bien plus souvent un ensemble de classes) visant à ce que l'utilisateur puisse manipuler ses tables de données comme si c'étaient des objets avec en plus une abstraction à l'accès aux données, ainsi que des fonctions que le développeur n'a plus ou pas à gérer et qui lui facilitent la vie :

- Connexion à la base
- Instanciations des objets
- Mises à jours etc...

On s'affranchit ainsi de beaucoup de traitements et c'est appréciable !

Doctrine est l'ORM intégré au framework PHP Symfony2, figurant aussi parmi les ORM les plus connus qui existent actuellement. Il est utilisé dans d'autres frameworks très connus tels que Zend Framework, et est aussi simple à prendre en main que puissant.

3. Twig

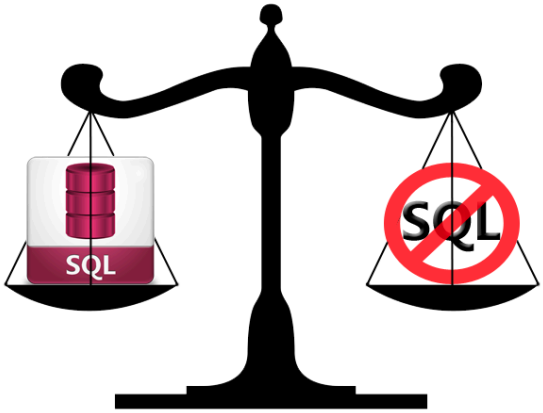
L'une des raisons qui nous a poussée à utiliser le framework PHP symfony est l'existence du moteur de templates du nom de Twig.

Twig est un moteur de template PHP directement intégré dans Symfony2. Très puissants, les templates Twig sont conçus pour être simples et ne traitent aucun code PHP. De par sa conception, le système de template Twig s'occupe de la présentation, pas de la logique. Il permet ainsi de séparer les couches de présentation et les couches métiers.

Twig peut aussi faire des choses que PHP ne pourrait pas faire, comme le contrôle d'espaces blancs, le bac à sable, l'échappement de caractères automatiques et contextuels et l'inclusion de fonctions et de filtres personnalisés qui n'affectent que les templates. Twig contient de petites fonctionnalités qui rendent l'écriture de templates plus facile et plus concise.

Gestion de la base de donnée

Pour notre Système de Gestion de Base de Donnée, nous avons eu à faire un choix entre les systèmes **SQL** et **NoSQL** (Not Only SQL). Si MySQL est le plus utilisé historiquement, NoSQL (Not Only SQL) se développe dans le monde du web. Pour la plupart des développeurs, MySQL est déjà une technologie connue et NoSQL semble représenter un « Eldorado » de par sa légèreté et surtout parce qu'il aurait été pensé pour le web.



D'une part, MySQL est le plus connu et utilisé des SGBD. Il repose sur le modèle relationnel : des tables ont des enregistrements , et ces tables peuvent avoir des relations. Ceci a l'avantage de pouvoir lier très facilement des enregistrements d'une table à l'autre. De plus, lors des enregistrements, les transactions sont soumises aux contraintes ACID (atomicité, cohérence, isolation et durabilité), ce qui signifie qu'un enregistrement incomplet ou incorrect ne sera pas enregistré en base. De quoi sérieusement limiter les erreurs ! MySQL permet ainsi de facilement structurer les informations et de les réutiliser avec aisance.

De l'autre, plutôt que de s'appuyer sur des contraintes lourdes et consommatrices de ressources, NoSQL se concentre sur l'accès direct à l'information. Inspiré tout d'abord des annuaires, NoSQL cherche donc à mettre en avant la lecture des données avant tout.

Ainsi, en pesant le pour et le contre, notre choix s'est porté sur une base de données **MySQL**. Pourquoi cela ?
Pour la simple et bonne raison que, bien que NoSQL permette un accès rapide aux données, dans notre projet les utilisateurs auront besoin de stocker de grandes quantités de données (recettes) pour les réutiliser de temps à autre et MySQL semble être le plus adapté pour cela. De plus, MySQL continue d'évoluer et justement dans le sens de la scalabilité et la rapidité d'accès en lecture. Nous n'avons donc pas nécessairement besoin de passer par le NoSQL pour avoir de bonnes performances.

Gestion des révisions : Git

Afin de nous faciliter le développement de ce projet et surtout de stocker l'évolution du code source nous avons décidé de nous servir d'un logiciel de gestion de versions : Git.

Un logiciel de gestion de versions est un logiciel qui permet de suivre les différentes versions d'un ensemble de fichiers. Il permet de retrouver un ou des fichiers tels qu'ils étaient à une date donnée. Plus précisément, le développeur demande manuellement, et aussi souvent qu'il le désire, au logiciel de gestion de version de stocker l'état de ses fichiers à un instant donné, lui permettant de revenir à cette version à tout moment.

Travaillant en binôme sur ce projet, le choix de l'utilisation d'un logiciel de gestion de version est judicieux dans la mesure où il permet de faciliter le travail entre plusieurs personnes. Chacun de nous pourra ainsi travailler sur une copie des sources sur son propre ordinateur, puis transmettre les versions stables qui seront récupérées sans conflit.

Structure des données