Projet 13 : Projet personnel

Comment j'ai implémenté les 12 bonnes pratiques de l'Xtreme Programming

Comment j'ai implémenté les 12 bonnes pratiques de l'XP :

1_ Planning game :

Cette partie est plutôt applicable pour un travail en équipe, or j'ai travaillé toute seule pour ce projet.

2_ Métaphores :
J'ai eu du mal à cerner ce point de l'Extreme Programming. Il me semble qu'il s'agit d'utiliser des mots et des notions compréhensibles par tous pour définir le système que l'on développe. Dans le cas de mon projet, s'agissant d'un système concret, je pense que sa description est dès le départ imagée et compréhensible par tous.

3_ Release fréquentes :

Mon projet étant subdivisé en fonctionnalités de taille raisonnable, il aurait été aisé de procéder à des releases fréquentes pour chaque ajout de fonctionnalité une fois l'architecture globale mise en place.
Cela apporte un cadre de développement sécurisé et il est agréable lors du développement de voir son application fonctionner au fur et à mesure.
Cependant, pour des questions de coût du serveur, j'ai fait des push qui auraient correspondu à une release sur le master repository Git, sans toutefois les mettre en production.

4_ Conception simple :

La préparation du projet par des spécifications claires et documentées ainsi que le suivi de l'architecture prônée par Django m'a permis une conception simple.

5_ Client sur site :

N'ayant pas de client, cette partie n'était pas applicable.

6_ Développement en binôme :

Pour un projet individuel, ce point est difficilement rempli. Néanmoins, dans mon travail en entreprise dans le cadre de l'alternance, j'ai été amenée brièvement à travailler en pair programming, et j'ai trouvé cette approche très intéressante et efficace.

7_ Rythme soutenable :

L'avantage de cette formation est que l'on avance à son rythme. Effectuant la formation en alternance, j'ai eu la possibilité de garder des congés pour la fin de mon alternance. Cela m'a permis d'avancer sans stress et sans me donner des objectifs inatteignables. Cette sécurité m'a évité de travailler dans la précipitation, et je pense que je n'en ai été que plus efficace.

Le fait d'avoir des fonctionnalités optionnelles à développer à la fin en fonction du temps m'a

également permis d'avoir un objectif adaptable à l'avancement de mon projet, et donc de ne pas travailler dans la précipitation: les fonctionnalités non implémentées ce jour seront implémentées dans une version ultérieure.

8_ Standards de code :

J'ai été habituée depuis le début du parcours à suivre les conventions python puis Django, donc ce point était assez évident. Je trouve qu'il permet à tous les développeurs de partager un code uniforme et facilement maintenable.

9_ Appropriation collective du code :

Partie difficilement applicable en développant en solo. Toutefois, j'ai tenter d'écrire un code le plus explicite possible, afin qu'il soit facilement compréhensible par une personne extérieure au projet.

10_ Tests :

Le plan de test a été réalisé en amont de la programmation. J'ai tenté une approche de Test Driven Programming. Pour la partie centrale du projet et ce qui était le plus simple, c'est ce que j'ai réalisé. Par contre, à la fin du projet, où je m'engageais vers des choses moins connues comme les graphes, j'ai procédé au développement avant de procéder aux tests.

11_ Intégration continue :

Le découpage du développement en petites releases et l'utilisation de Travis m'ont permis de procéder à une intégration continue pour ce projet.

12_ Refactoring :

Je suis particulièrement sensible à ce point, effectuant mon alternance dans un cadre où il n'est pas respecté : les évolutions et la maintenance du code s'en trouvent très complexifiés, voire impossibles. C'est donc pour moi avec plaisir que j'ai tenté de refactoriser le code du projet losque cela s'avérait pertinent. Je suis toutefois consciente que certaines parties sont encore à améliorer.