Le projet SOSIE (Système d'Optimisation pour les Services Internes à un Etablissement) a été initié dans le cadre du cours d'architecture logicielle afin de permettre aux étudiants de collaborer ensemble via Github sur un même projet.

Organisation du travail et des équipes

Pour ce faire des équipes ont été constitué (3 étudiants à 4 étudiants par équipe). Pour chaque équipe, un chef d'équipe est désigné, son rôle est d'évaluer le travail à faire, répartir les tâches, s'assurer que chaque membre de l'équipe dispose des compétences nécessaires pour accomplir ses tâches en respectant la deadline qui serait idéalement le dimanche.

Chaque semaine une équipe se porte volontaire pour travailler en dépôt local sur un sprint du projet (que ce soit sur la partie développement ou support). A la fin du sprint, le chef d'équipe effectue un *pull-request* pour demande au propriétaire du dépôt (le prof) de merger leur travail. Avant cela, le chef de l'équipe doit s'assurer qu'il n'ya pas de conflits de merge suite à la fusion des différents codes. Il est donc de sa responsabilité de faire les vérifications et les tests nécessaires à la validation du code. Aussi, les messages de commit du sprint doivent être parlants et clairement énoncé.

Après validation du pull-request, les membres de l'équipe présente chaque début de semaine au reste des équipes le travail réalisé (ce qui a été fait et potentiellement ce qui reste à faire pour le sprint) et passe la main à une autre équipe pour entamer le sprint suivant. Suite à la présentation du travail réalisé, toutes les équipes pourront tester le code

Convention de rédaction

Norme à respecter pour rédiger ses messages de commit :

Sprint X – Thématique – Éléments concrets

Situations particulières

Si une équipe ne finit pas son sprint à temps, un délai supplémentaire de 24 h peut lui être accordé. Toutefois si au bout de 2 jours le sprint n'est toujours pas fini, une réorganisation du travail est nécessaire.
Dans ce cas un rallongement de temps sera fait pour permettre à l'équipe suivante de reprendre le sprint et le terminer avant de s'attaquer à un autre sprint.
Où dans quel cas, permettre à l'équipe de terminer le sprint en cours tandis qu'une autre équipe démarre un nouveau sprint qui lui est indépendant du sprint précédent.

Organisation des fichiers

Au niveau de l'organisation des fichiers du code source, il y va de la responsabilité de l'équipe de développement. Cependant, le Framework « ***Spring*** » exige une certaine arborescence des fichiers (il n'est donc pas possible de les déplacer au risque de générer des erreurs).

Pour ce qui concerne la documentation du projet, nous avons 3 niveaux

- ✓ 1 répertoire **/Documentation** qui comporte l'ensemble des documents relatifs au projet
  - BienCommencer.pdf document d'installation et de configuration du projet
  - ConventionsDeCodage.pdf qui présente l'ensemble des règles à suivre pour uniformiser le code (règle de nommage des packages, des classes, des méthodes des variables et des constantes ; les commentaires et la documentation du code source ; l'écriture des instructions ainsi que le style d'indentation)
  - Sprint-X.pdf qui explique de façon détaillée tout le travail réalisé sur le sprint X
  - Sprint-X-oral.pdf doc document support pour la présentation orale des équipes
  - Methodologie.pdf qui présente l'organisation du projet (fonctionnement des équipes, organisation du travail …)
  - Un dossier modifiables où figure les fichiers modifiables (.odt .doc .docx)

- ✓ **LICENCE** pour indiquer le type de licence utilisé pour le projet
- ✓ **README.md** qui présente le projet, les équipes, l'état du projet , une description rapide des options et la configuration requise pour utiliser le projet et des instructions claires sur comment builder le projet
- ✓ **SPRINT.md** qui présente le sprint en cours (période, objectifs à réaliser, contraintes, équipe) ainsi qu'un historique des sprints passés