Rapport de Gestion de Projet : Développement d'un Logiciel de Todo-List en Python

Chef de Projet / Product Owner : Hugo Grigord

Scrum Master : Kevin Lang

Membres de l'équipe : Jules Goetz, Marius Breinlen, Zackarie Assens, Swan Marcuzzi, Maxime Brodin, Haris Uka

Introduction

Ce rapport décrit le déroulement du projet de développement de l'application Taskflow, une application de gestion de tâches collaboratif réalisé dans le cadre de la SAÉ 5.02. Nous avons appliqué les principes de la méthodologie SCRUM et utilisé des outils collaboratifs modernes pour garantir une gestion de projet efficace, une communication fluide et un suivi rigoureux des tâches.

Gestion de Projet avec SCRUM

Organisation de l'équipe :

- **Product Owner (Chef de Projet)** : Supervise la vision globale et assure la priorisation des tâches dans le backlog.
- **Scrum Master** : Facilite les processus SCRUM et élimine les obstacles à la progression.
- **Développeurs** : Responsables de la réalisation des fonctionnalités spécifiques.

Rôles assignés :

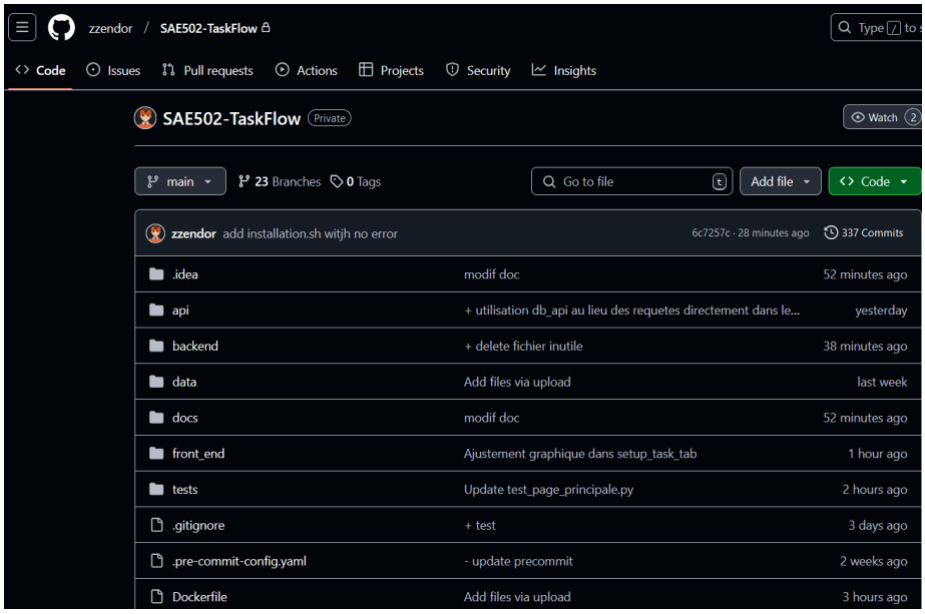
- **Jules** : Gestion de la base de données. Développeur Back-End. Coordination et développement des relations Front-End /Back-End.
- **Marius** : Développeur Back-End. Coordination et développement des relations Front-End /Back-End.
- **Zack & Swan** : Développeurs Front-End.
- **Kevin** : Gestion des droits et des priorités des utilisateurs.
- **Maxime** : Développeur Back-End
- **Haris** : Supervision de la documentation/ Développeur Back-End
- **Hugo** : Développement des relations Front-End / Back-End. Mise en place du Docker. Développeur Back-End

Outils Utilisés

1. GitHub

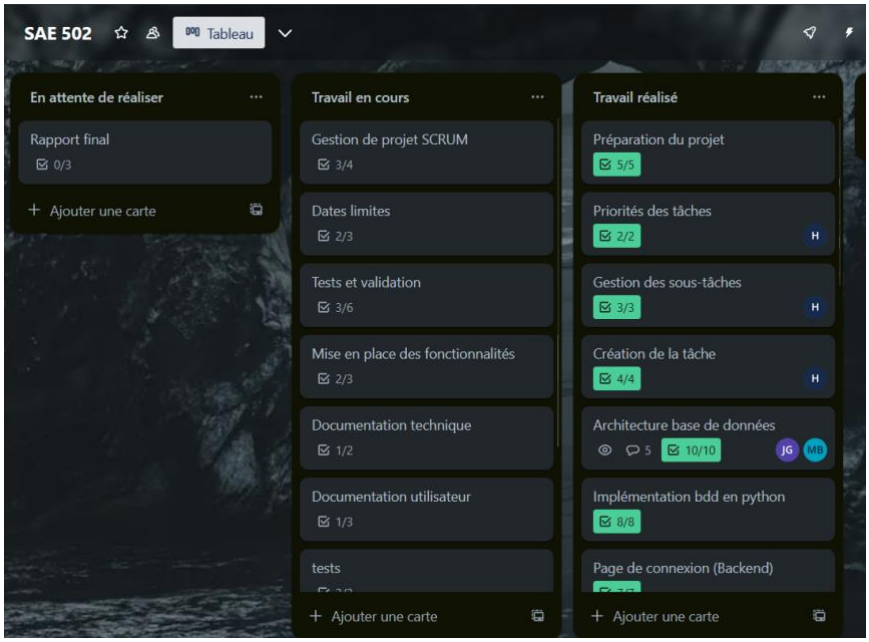
- Dépôt centralisé pour le suivi des versions et la collaboration.
- Utilisation des branches pour des fonctionnalités spécifiques, avec validation par pull requests et revues de code.
- Actions GitHub pour les tests automatisés.
- Répartition équitable des commits reflétant une contribution collective

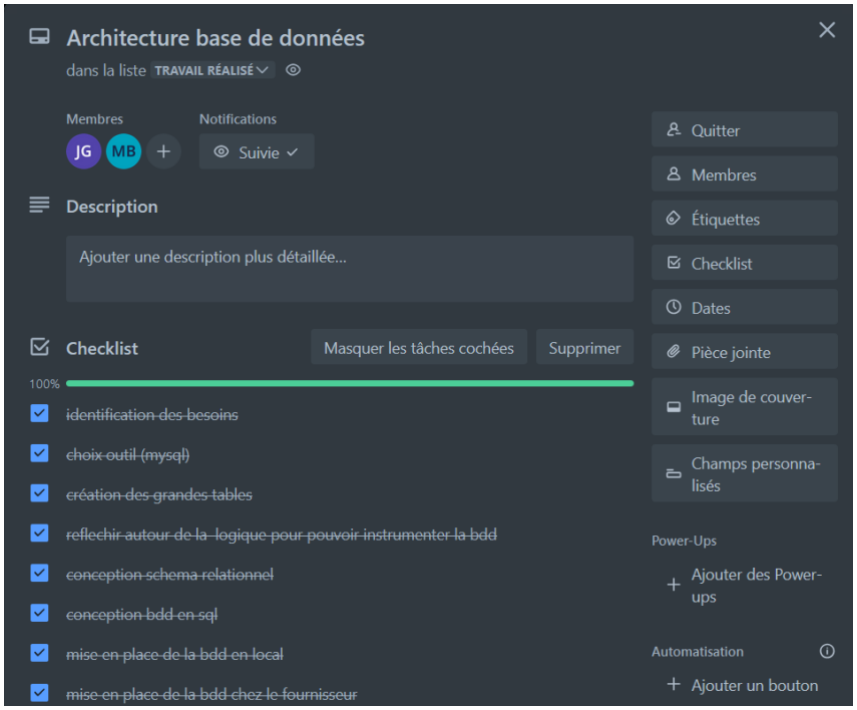
.



2. Trello

- Chaque membre a géré des cartes correspondant à ses responsabilités, avec des mises à jour régulières des statuts.
- Ajout d'étiquettes pour prioriser les fonctionnalités critiques (par exemple : "urgent", "amélioration").





3. Discord

- Communication en temps réel.
- Réunions journalières pour des points rapides (daily stand-up) afin de synchroniser les progrès et signaler les obstacles.
- Canaux dédiés par thématique (Base de données, Front-End, Back-End, Gestion des utilisateurs).

Fonctionnement Collaboratif et Répartition des Tâches

Nous avons adopté un cycle itératif SCRUM avec des sprints hebdomadaires. Chaque sprint comprenait :

1. **Planification des tâches :** Identification des fonctionnalités à développer et priorisation dans le backlog.
2. **Exécution :** Développement par les membres selon leurs rôles définis.
3. **Revue :** Validation des fonctionnalités via des démonstrations.
4. **Rétrospective :** Analyse des réussites et des améliorations possibles pour le sprint suivant.

Exemples concrets :

- **Jules** : Création et gestion de la structure de la base de données avec des modèles adaptés aux tâches et sous-tâches, création des fichiers db_requests et db_api permettant des interactions simples avec la base de données. Liaison du Front-End et du Back-End (Affichage des utilisateurs dans un projet, gestion des invitations des utilisateurs dans un projet). Tests unitaires pour la partie base de de données.

- **Marius** : Aide à la création de la base de données avec Jules. Liaison du backend à la base de données qui marchait uniquement en local. Liaison du Front-End et du Back-End (création de l'utilisateur, création, suppression et modification des tâches/sous-tâches, ajout et suppression de projets, affichage des tâches d'un projet). Rédaction de la documentation technique pour le fichier pageprincipale.py.

- **Maxime** : Mise en place de l'authentification des utilisateurs, intégration de l'API Google (mails, token). Tests unitaires pour la partie authentification. Création de la documentation technique (Docstring pour le backend et documentation complète avec Sphinx).

- **Hugo** : Développement des relations Front-End / Back-End. Mise en place de l'environnement Docker pour conteneuriser l'application. Marquage des tâches et sous-tâches comme terminées, création de nouvelles fonctions pour la gestion des informations liées aux comptes.

- **Zack** : Création de la structure Front-End globale, notamment pour l'authentification. Ajout d'éléments graphiques à la demande de l'équipe. Design des logos et de l'interface. Liaison entre le back-end et le front-end. Debug et rédaction de la documentation pour la page d'authentification.

- **Swan** : Développement du Front-End de la page principale (to-do list) sans base de données, juste l'affichage graphique. Aide au développement de la page d'authentification. Rédaction des guides d'installation (classique ou Docker) et documentation d'utilisation détaillée. Amélioration du README.md avec des instructions claires pour l'installation et un aperçu des fonctionnalités. Debug et gestion des demandes de l'équipe.

- **Haris :** Rédaction du fichier README pour le dépôt GitHub, de la première version du guide d'utilisation, ainsi que du rapport de gestion de projet détaillant les méthodes employées.
  Implémentation des fonctionnalités de création de tâches et de sous-tâches. Développement des fonctions pour l'envoi de rappels par e-mail via un compte Google et mise en place d'une fonctionnalité d'invitation d'utilisateur à un projet.

Résultats Obtenus

1. **Logiciel fonctionnel** :
   - Création, modification et gestion des tâches/sous-tâches.
   - Affectation des tâches à des utilisateurs spécifiques.
   - Ajout de dates limites, priorités
   - Système d'authentification et gestion des droits

2. **Image Docker** : Fourniture d'un environnement prêt à l'emploi pour exécuter l'application.

3. **Documentation complète** :

- Guide utilisateur.
- Guide installation.
- Documentation technique (générée avec Sphinx).

Analyse de la Gestion de Projet

Points forts :

- Communication fluide via Discord, évitant les retards.
- Suivi rigoureux sur Trello, permettant une vue d'ensemble claire des tâches.
- Dépôt GitHub bien structuré, facilitant la collaboration et l'intégration continue.

Défis rencontrés :

- Problèmes d'intégration Front-End/Back-End rapidement résolus grâce aux réunions de coordination.
- Gestion des disponibilités des membres pendant certaines semaines

.

Leçons apprises :

- Importance d'une planification précise au début de chaque sprint.
- Nécessité de rédiger des tests unitaires dès le début du développement.

Conclusion

Ce projet nous a permis d’appliquer concrètement les principes de la gestion de projet informatique et de SCRUM. Grâce à une utilisation optimale des outils collaboratifs (GitHub, Trello, Discord), nous avons atteint les objectifs fixés tout en respectant les délais.

Nous avons également amélioré nos compétences techniques et organisationnelles, ce qui constitue une expérience précieuse pour nos futurs projets professionnels.