Avenue du Ciseau, 15
1348 Ottignies-Louvain-la-Neuve

Développement d'un système d'information pour la société Master Services

Travail de fin d'études

JAUJATE OULDKHALA Ikram

Rapporteur :
Monsieur S. Buys

Année académique 2021-2022

Table des matières

1 Introduction

1.1 Contexte général

1.1.1 Client

Le client est une entreprise dans secteur du bâtiment et travaux publics et est situé à Sint-Pieters-Leeuw. Dans le cadre de ses activités, le client doit constamment gérer le flux de matériaux, ses clients, ses employés, ses fournisseurs ainsi que toutes les facturations qui en découlent.

Le client ne possède actuellement aucun moyen informatisé de gérer l'ensemble de ses entités et m'a demandé de concevoir une solution pour répondre à l'ensemble de ses besoins :

— Gestion clients

— Gestion des Projets

— Gestion Matériel

— Gestion Stock

— Gestion Main d'Oeuvre

— Gestion du personnel

— Gestion utilisateurs

— Gestion Factures

— Gestion Devis

1.2 Objectifs

Cette solution doit être conçue de telle sorte qu'elle puisse être adaptée au fil des années en fonction de l'évolution des besoins du client. De plus, le client étant souvent sur chantier, il est primordial que l'application puisse être facilement portable d'un système / ordinateur à un autre. Il a donc été décidé en commun accord avec le client que la solution sera déployée sous forme d'une application web ce qui apportera une grande flexibilité au niveau de l'utilisation de la solution.

Le volume d'information à gérer étant conséquent, la visualisation doit principalement être adaptée **aux écrans d'ordinateur.**

Je pense que ce projet rentre tout à fait dans le cadre d'un TFE étant donné qu'il require l'analyse, le développement, le déploiement et la maintenance d'une application web à des fins d'apporter une solution adéquate répondant aux spécificités d'un client. Le développement se fera à l'aide de technologies modernes qui me permettront d'offrir une interface au goût du jour et modulable. À des fins d'améliorer la productivité, je ferais usage de multiples techniques DevOps telles que le déploiement continu et bien d'autres.

2 Méthode de travail

2.1 Méthodologie

Vu l'envergure du projet et de tâches techniques à accomplir, la méthodologie que j'ai choisie pour la réalisation de ce projet est méthodologie Agile. Spécifiquement j'ai décidé de travailler avec la méthode Scrum avec son organisation en sprints.

Tout d'abord, j'ai pu définir avec le client les principales fonctionnalités à intégrer dans le projet. Ces fonctionnalités sont classées par ordre de priorité et du temps estimé à la réalisation des celles-ci.

En outre, les principales fonctionnalités contenues dans le projet sont détaillées. S'agissant d'un projet de grande envergure, chacune des principales fonctionnalités a été divisée en petites user stories qui me permettront d'avoir un produit livrable au client à la fin de chaque sprint. Les sprints auront **une durée d'environ 2 semaines.**

À la fin de chaque sprint, un délivrable correspondant aux tâches / user stories effectuées lors du sprint devra être présenté au client. Ce dernier devra alors vérifier et valider les différentes tâches effectuées.

L'avantage de cette méthode est que, en cas d'erreurs et/ou non validation des tâches de la part du client, ces dernières pourront être revues et corrigées pour le prochain sprint.

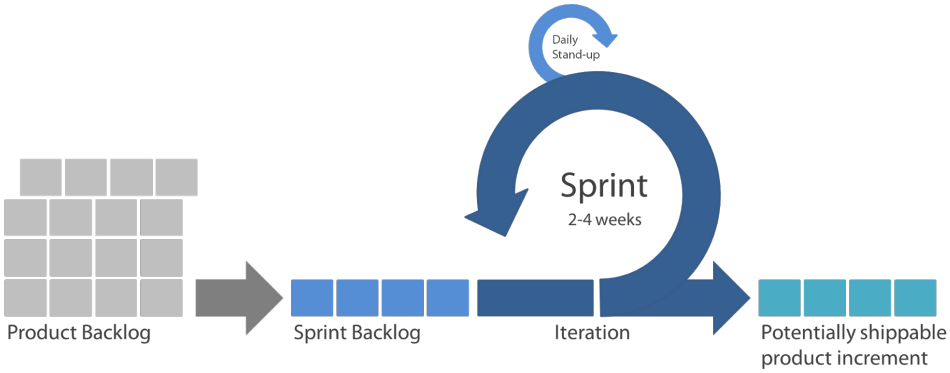

FIGURE 1 – *L'implementation de la méthode Scrum* de Anna Pérez

Pour l'organisation des tâches à effectuer pour chaque sprint, j'utiliserai l'outil Trello. Cet outil me permettra de diviser les tâches à effectuer pour chaque sprint, ce qui facilitera la visualisation de l'avancement du projet.

Cette méthode de travail permet donc de ne pas définir certaines user stories qui ne seront peut-être jamais mises en place.

2.1.1 Gitflow

J'ai décidé de travailler avec le gitflow par branche qui me permettra d'avoir une division au niveau des fonctionnalités qui seront implémentées lors du projet.

— une branche 'develop' qui correspond à la branche 'master' du 'github-flow'

— les releases sont préparées sur une branche spécifique (fusion depuis la branche 'develop' jusqu'à ce que la release soit validée)

— lorsqu'une release est prête, elle est fusionnée sur la branche 'master'

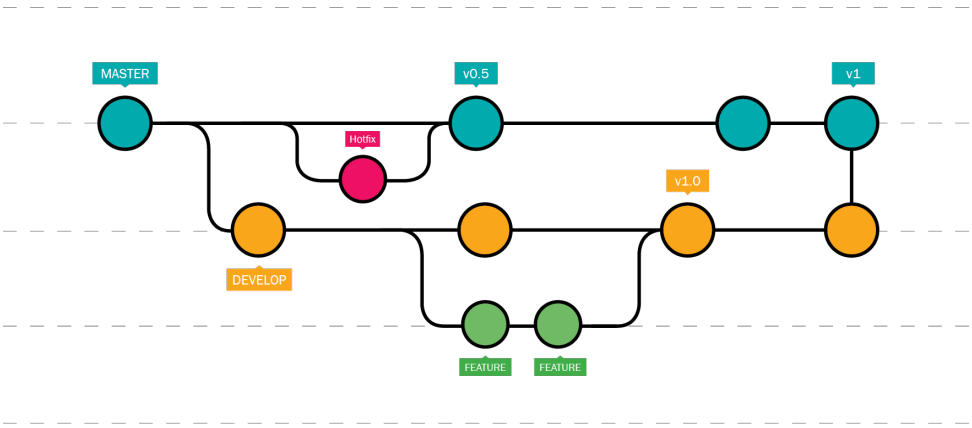

FIGURE 2 – *Gitflow Strategy* de Atlassian

2.2 Choix des technologies

2.2.1 Frontend

En ce qui concerne la technologie frontend, j'ai décidé d'utiliser React car elle me permet de créer des interfaces utilisateur ou des composants d'interface utilisateur rapides et interactifs pour les utilisateurs d'applications web et mobiles.

1. **DOM Virtuel** : Il permet de générer le DOM ("Document Object Model", structure des éléments qui sont générés dans le navigateur web lors du chargement d'une page) de manière dynamique, ce qui nous permet de visualiser les changements de données sans devoir recharger/render à nouveau la page entière, mais seulement le composant qui a été mis à jour.
2. **Grande communauté** : Il est soutenu par une large communauté, ce qui nous permet d'avoir un grand nombre de libraires disponibles.
3. **Composants réutilisables** : Elle est constituée de composants qui sont réutilisables, ce qui rend l'application plus évolutive et plus facile à maintenir car les erreurs se produisent dans la fonctionnalité du composant lui-même.

Ces avantages permettent d'améliorer l'expérience de l'utilisateur lors de la navigation dans l'application web, la rapidité du chargement des pages et facilitent la maintenance de l'application.

2.2.2 Web Server

Au niveau du serveur web, mon choix se porte sur Caddy Server et ce pour plusieurs raisons

1. **Simple** : Possède une configuration très simple qui nous permettra de le mettre en place en quelques minutes.
2. **HTTPS par défaut** : Utilise Let's Encrypt pour mettre le site en HTTPS complet automatiquement, sans aucune configuration et le renouvellement des certificats SSL/TLS se réalise de manière automatique.
3. **Multiplateforme** : Il est multiplateforme et je serais en mesure d'exécuter Caddy directement par le biais de Docker, ce qui rendra sa mise en œuvre encore plus facile.

2.2.3 Backend

Comme pour le frontend, le choix du backend est également essentiel. Dans ce cas, j'ai décidé de travailler avec Node.js et ce pour plusieurs raisons.

1. **Très rapide** : Les tâches courantes comme la lecture ou l'écriture dans la base de données sont exécutées rapidement et il capables de gérer des connexions simultanées à haut débit
2. **Grande communauté** : Il est soutenu par une large communauté, ce qui nous permet d'avoir un grand nombre de libraires disponibles.
3. **MVC** : Permet de travailler en MVC, ce qui permet une structure correcte du code.
4. **Asynchrone** : Étant un système asynchrone, il permet d'accélérer les applications web. Il est capable d'envoyer gros volumes de données sans bloquer le serveur qui reste ainsi disponible pour traiter d'autres tâches.
5. **compatible** : Permet un développement multiplateforme qui est axé sur tous les types d'appareils et de plateformes d'OS (iOS, Android, desktop et web). Le code est réutilisable et entièrement compatible avec tous les principaux systèmes d'exploitation, notamment Linux, Windows, ainsi que macOS, ce qui va nous permettre de rendre notre Web Application accessible depuis toutes les plateformes.

2.2.4 Database

Mon choix pour la base de données est PostgreSQL pour plusieurs raisons :

1. **DB Relationnelle** : Puisque les données doivent être ordonnées et structurées et que des relations doivent exister entre les différentes données, il est essentiel d'utiliser une base de données SQL afin de garantir l'organisation de ces dernières
2. **SQL** : PostgreSQL utilise le langage SQL, qui est le langage le plus utilisé pour les bases de données relationnelles.
3. **Compatible** : PostgreSQL est entièrement compatible ACID. ACID est un acronyme pour Atomicité, Cohérence, Isolation et Durabilité. Il garantit donc que les transactions n'interfèrent pas entre elles. Cela garantit les informations contenues dans les bases de données et la pérennité des données dans le système.
4. **Hot-Standby** : Il dispose de l'option Hot-Standby qui permet aux utilisateurs d'accéder aux tables en mode lecture pendant que les processus de sauvegarde ou de maintenance sont en cours.

PostgreSQL jouit d'une solide réputation en matière de fiabilité, de robustesse des fonctionnalités et des performances.

2.3 Autres

2.3.1 API Documentation

2.3.2 Linter

3 User Stories

3.1 Description

3.1.1 Exemple

3.2 Liste des fonctionnalités dévélopées

4 Déploiement

4.1 Études des différentes solutions

4.1.1 Heroku

4.1.2 Serveurs dédiés OVH

4.1.3 Justification du choix

4.2 Intégration Continue et Déploiement Continu

4.2.1 Docker

4.2.2 Scripts

4.2.3 Github workflow

5 Testing

5.1 Unitaires

5.2 Integrations

5.3 End-to-End

6 Sécurité

6.1 Frontend

6.1.1 Routage

6.2 Backend

6.2.1 Routage API

6.2.2 Base de données

6.2.3 En-têtes HTTPS

6.3 Web Server

6.3.1 SSH

6.3.2 Firewall

6.3.3 Reverse-proxy

6.4 Autres

6.4.1 Libraries

7 Monitoring

8 Conclusion

9 Bibliographie

[1] Smashing Magazine (2020, 2 mai), sur le site *Implementing Dark Mode In React Apps Using styled-components* Consulté le 20 janvier 2022
`https://www.smashingmagazine.com/2020/04/dark-mode-react-apps-styled-components/`

[2] Atlassian (2021, 22 mars), sur le site *Beautiful and accessible drag and drop for lists with React* Consulté le 22 janvier 2022
`https://github.com/atlassian/react-beautiful-dnd`

[3] User Story (2019), sur le site *User story - Wikipedia* Consulté le 24 novembre 2021
`https://en.wikipedia.org/wiki/User_story`

[4] Stack Overflow (2020, 10 décembre), sur le site *Upgrading Node.js to latest version* Consulté le 22 décembre 2021
`https://stackoverflow.com/questions/10075990/upgrading-node-js-to-latest-version`

[5] Tang, R. (2020, 10 décembre), sur le site *How to install Node JS and NPM* Consulté le 14 décembre 2021
`https://www.makersupplies.sg/blogs/tutorials/how-to-install-node-js-and-npm-on-the-raspberry-`

[6] Terzi, R(2018, 16 novembre), sur le site *Qu'est-ce que l'approche CI/CD ?* Consulté le 22 janvier 2022
`https://www.redhat.com/fr/topics/devops/what-is-ci-cd`

[7] Shadow-M-P (14 mai 2021), sur le site *Méthode agile — Wikipédia* Consulté le 23 janvier 2022
`https://fr.wikipedia.org/wiki/Méthode_agile`

[8] PostgreSQL Tutorial (sans date), sur le site *PostgreSQL Transactions* Consulté le 3 décembre 2021
`https://www.postgresqltutorial.com/postgresql-transaction/`

[9] PostgreSQL Tutorial (sans date), sur le site *PL/pgSQL For Loop* Consulté le 14 décembre 2021
`https://www.postgresqltutorial.com/plpgsql-for-loop/`

[10] PostgreSQL Tutorial (22 mai 2021), sur le site *PostgreSQL CREATE PROCEDURE* Consulté le 3 janvier 2022
`https://developer.mozilla.org/fr/docs/Web/HTTP/Headers/X-Frame-Options`

[11] Großgarten, G. (2021, 21 janvier), sur le site *Copy a folder to a remote server using SSH* Consulté le 10 novembre 2021
`https://github.com/garygrossgarten/github-action-scp`

[12] OVH (sans date), sur le site *Qu'est-ce qu'un VPS ? Découvrez les avantages d'un VPS | OVHcloud* Consulté le 25 novembre 2021
`https://www.ovhcloud.com/fr/vps/definition/#:~:text=Grâce%20au%20VPS%2C%20vous%20profitez,en%20payant%20le%20juste%20prix.`

[13] Adaobi, A. (2021, 27 octobre), sur le site *Making HTTP Requests in Node.js with node-fetch* Consulté le 25 novembre 2021
`https://stackabuse.com/making-http-requests-in-node-js-with-node-fetch/`

[14] Laurent Hercé (2020, septembre 15), sur le site *Conteneurisation informatique : définition, avantages, différence virtualisation, solutions | Appvizer* Consulté le 26 décembre 2021
`https://www.appvizer.fr/magazine/services-informatiques/virtualisation/conteneurisation-informatique`

[15] Stack Overflow (2011, 3 juillet), sur le site *Can I use return value of INSERT. . . RETURNING in another INSERT ?* Consulté le 26 décembre 2021
https://stackoverflow.com/questions/6560447/can-i-use-return-value-of-insert-returning-in-ano