Jérémy Gfeller
Alexandre Junod

Table des matières

1 Analyse préliminaire

1.1 Introduction

Le projet est réalisé dans le cadre du CPNV. Deux professeurs qui sont, M. CHEVILLAT et M. HURNI se sont fait passer pour des clients qui nous demandent de réaliser une application pour eux.

Le but de ce travail est de se familiariser avec la gestion de projet ainsi que les processus à faire pour un projet. Nous avons dû créer un cahier des charges afin qu'ils sachent ce que nous allions développer, ce qui nous permet de savoir si nous avons bien compris leurs demandes.

Les clients nous ont demandés de créer une application qui permettra de rechercher dans un dossier spécifique un ou plusieurs fichiers. Des méthodes de tris doivent être implémentées. Une recherche de mot-clé dans un fichier, une date de modification, un auteur ou encore un nom de fichiers, seront les tris de notre application.

Cette application se fera à l'aide de Visual Studio 2017, avec le langage de programmation C# (se prononce « si charp »).

1.2 Organisations

Clients :

- M. Jérôme CHEVILLAT, jerome.chevillat@cpnv.ch
- M. Pascal HURNI, pascal.hurni@cpnv.ch

Elèves :

- M. Alexandre JUNOD, alexandre.junod@cpnv.ch
- M. Jérémy GFELLER, jeremy.gfeller@cpnv.ch

1.3 Objectifs

Ci-dessous vous trouverez les objectifs personnels du groupe :

- Se familiariser avec le langage de programmation de Visual Studio.
- Atteindre tous les objectifs demandés par les clients.

Les objectifs des clients sont les suivants.

L'utilisateur pourra utiliser l'application pour, rechercher des fichiers en donnant :

- Un mot clé
- L'auteur
- Une date
- Un nom ou une extension de fichier

Il aura la possibilité :

- D'ouvrir le document sélectionné
- De se rendre dans le dossier où la recherche s'est effectuée

1.4 Planification initiale

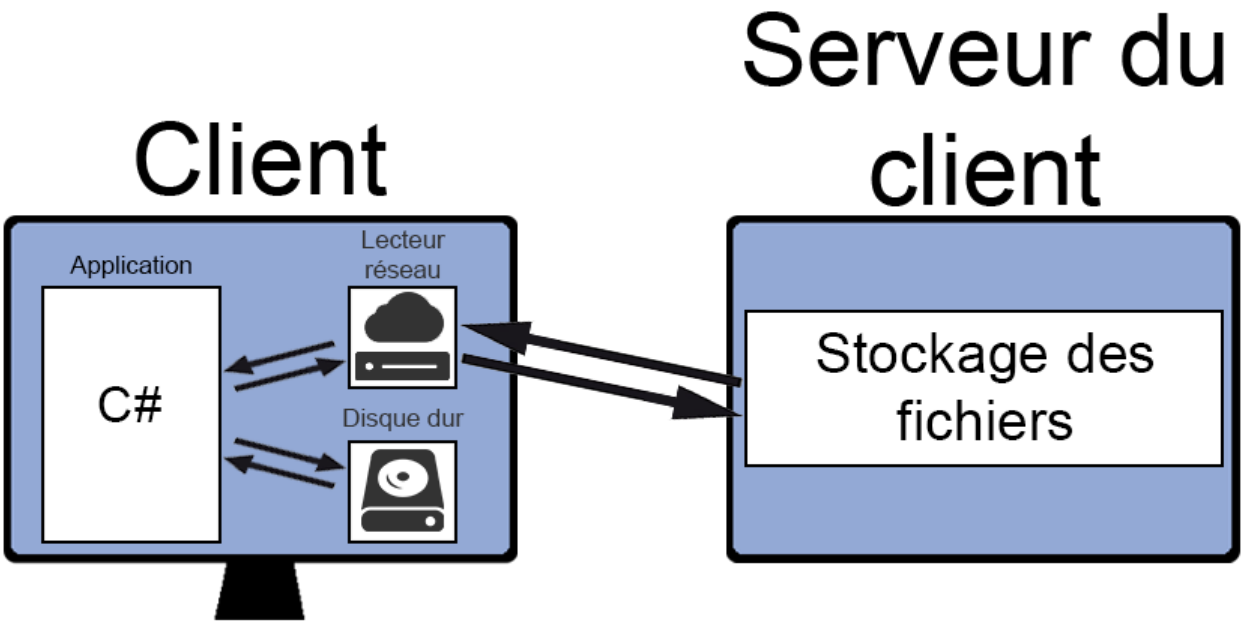
Le projet a débuté le 31 août 2018, date à laquelle nous avons reçu les consignes. Nous avons pu poser toutes les questions que nous voulions aux clients, grâce à un fichier Google Docs. La fin du projet est prévue le vendredi 14 décembre 2018. Une présentation du projet est prévue une semaine après le rendu.

Ci-dessous vous trouverez les sprints du projet.

	Description	Date de fin
Sprint 1	Analyse et conception du projet	14 septembre 2018
Sprint 2	Affichage de l'application	09 octobre 2018
Sprint 3	Tri des fichiers	09 novembre 2018
Sprint 4	Fonctionnalités supplémentaires	27 novembre 2018
Sprint 5	Correction de bugs et améliorations	14 décembre 2018
Clôture du projet	Livrer l'application aux clients	14 décembre 2018

2 Analyse / Conception

2.1 Vue d'ensemble



Voici un petit schéma qui explique comment l'application fonctionne et les actions qu'elle entreprend sur le lecteur réseau du client, ou sur son disque dur local. L'application fonctionne de la même manière si les données sont stockées en local ou non.

Lors du lancement de l'application, le client pourra choisir dans quel dossier il voudra faire la recherche. Une fois le dossier choisi, il peut ajouter et cumuler des critères de recherche pour trouver son fichier plus rapidement.

2.2 Stratégie de test

Les tests se feront au fur et à mesure qu'une fonctionnalité est implémentée. Nous allons la tester afin de contrôler qu'elle soit fonctionnelle et rapide.

Une fois que l'application arrivera au terme de son développement, nous laisserons sûrement des personnes de notre classe l'essayer pour voir si elle comporte des bugs ou autres.

Les tris fonctionnent séparément ou ensemble. L'utilisateur choisira s'il veut les combiner ou non. Une fois qu'un tri était implémenté, nous avons choisi un répertoire pour le tester et voir le temps que ça prenait pour afficher le contenu d'un répertoire.

2.3 Risques techniques

Pour les risques techniques, le fait de créer l'application sur Visual Studio en C# est un gros risque. La dernière fois que nous avions programmé avec cette technologie, nous étions encore en 2ème année CFC.

Nous devons nous remettre dans le bain au niveau du développement et se rappeler comment tout fonctionne sur Visual Studio.

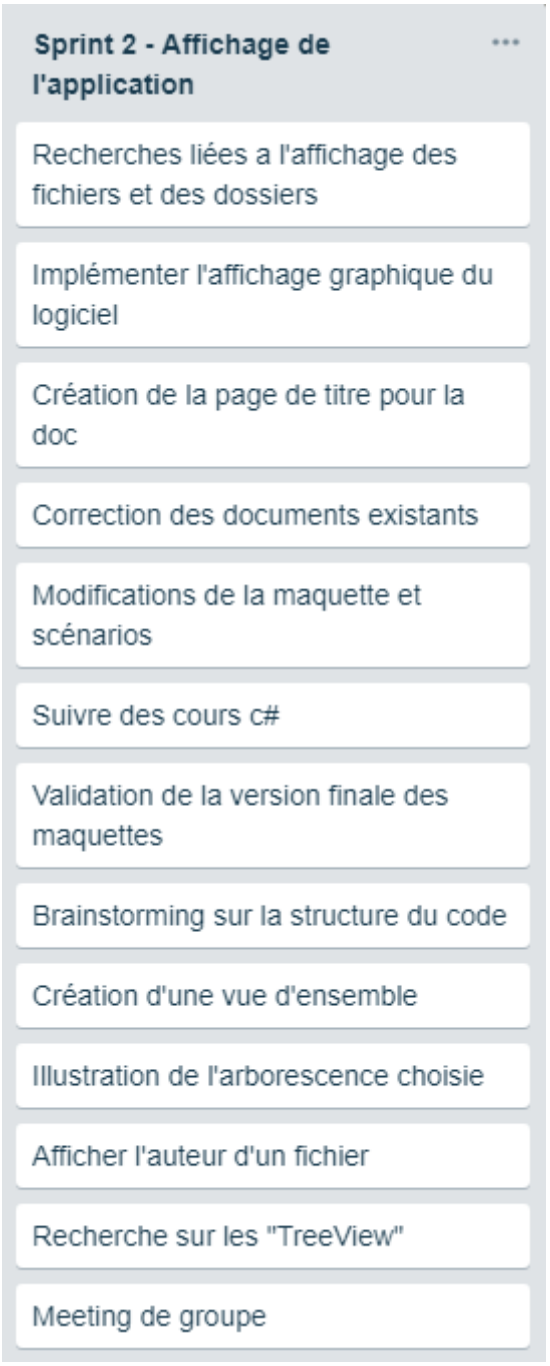
Afin d'arriver au résultat souhaité, nous avons utilisé des librairies permettant d'ouvrir des fichiers et de lire ce qu'il y avait à l'intérieur (PDF, Excel, Word, etc…). Suite à plusieurs tests, nous sommes venus à la conclusion que le temps de recherche était trop long, nous avons donc utiliser d'autre librairies qui permettent d'extraire le texte au lieu d'ouvrir le document, tel que Spire et iTextSharp.

2.4 Planification

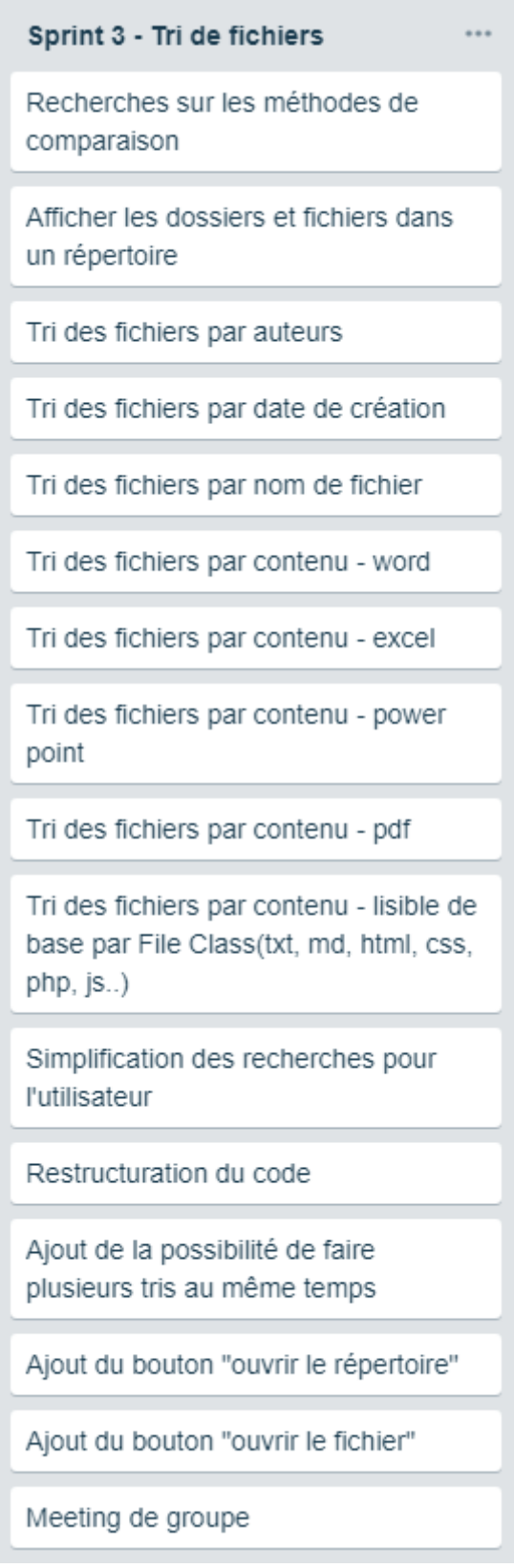
Ci-dessous vous trouverez les actions qui ont été entreprises durant le projet et dans chacun des sprints.

2.4.1 Sprint 1

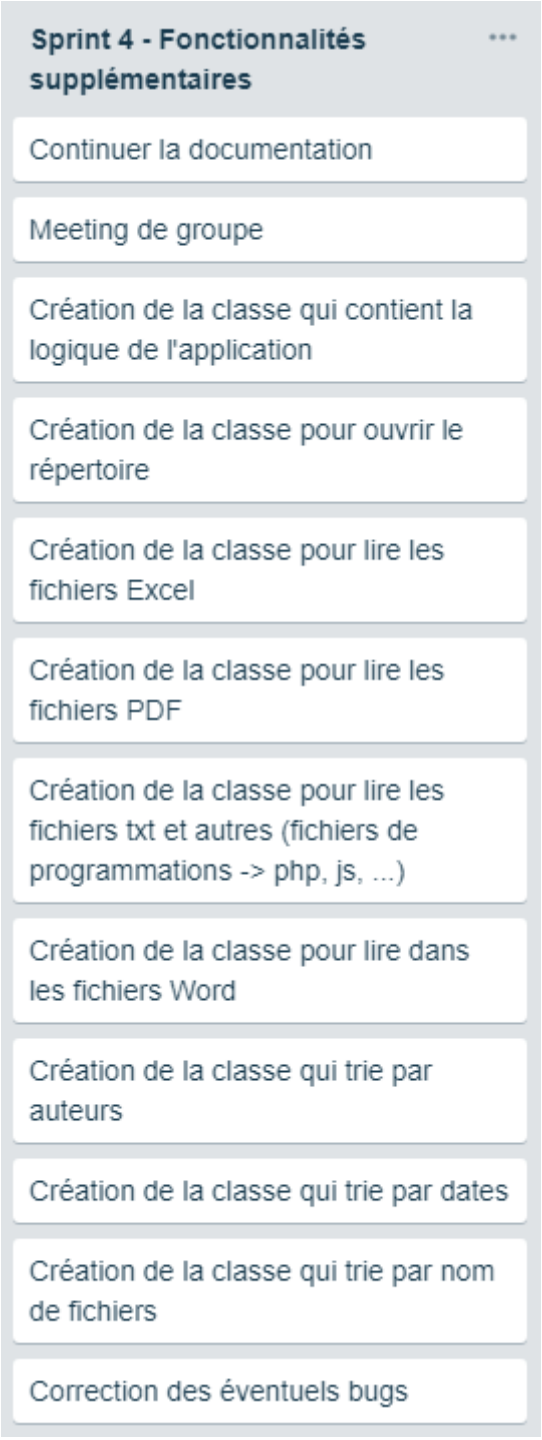
2.4.2 Sprint 2


Sprint 2 - Affichage de l'application
Recherches liées a l'affichage des fichiers et des dossiers
Implémenter l'affichage graphique du logiciel
Création de la page de titre pour la doc
Correction des documents existants
Modifications de la maquette et scénarios
Suivre des cours c#
Validation de la version finale des maquettes
Brainstorming sur la structure du code
Création d'une vue d'ensemble
Illustration de l'arborescence choisie
Afficher l'auteur d'un fichier
Recherche sur les "TreeView"
Meeting de groupe

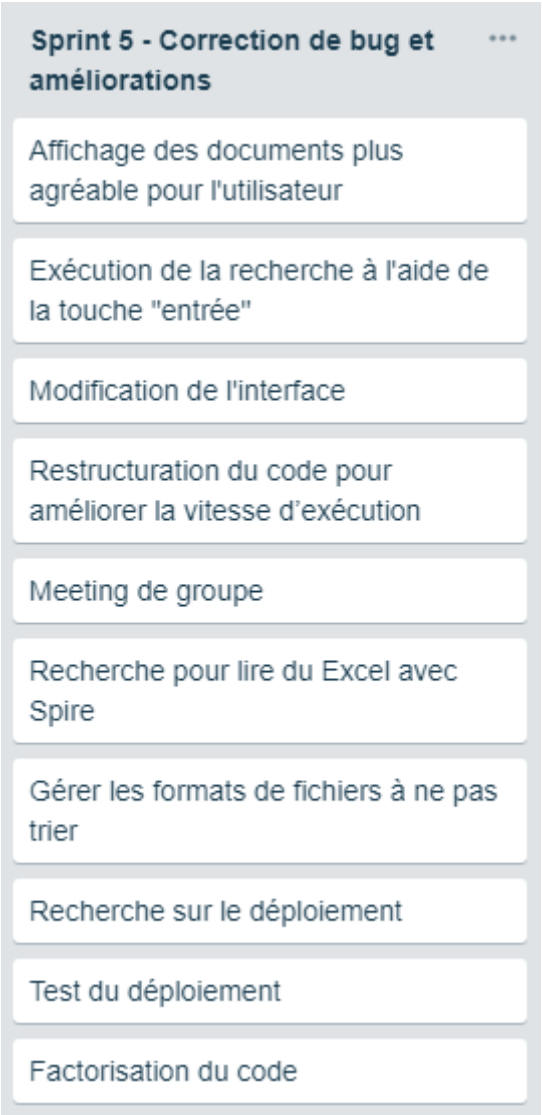
2.4.3 Sprint 3


Sprint 3 - Tri de fichiers
Recherches sur les méthodes de comparaison
Afficher les dossiers et fichiers dans un répertoire
Tri des fichiers par auteurs
Tri des fichiers par date de création
Tri des fichiers par nom de fichier
Tri des fichiers par contenu - word
Tri des fichiers par contenu - excel
Tri des fichiers par contenu - power point
Tri des fichiers par contenu - pdf
Tri des fichiers par contenu - lisible de base par File Class(txt, md, html, css, php, js..)
Simplification des recherches pour l'utilisateur
Restructuration du code
Ajout de la possibilité de faire plusieurs tris au même temps
Ajout du bouton "ouvrir le répertoire"
Ajout du bouton "ouvrir le fichier"
Meeting de groupe

2.4.4 Sprint 4


Sprint 4 - Fonctionnalités supplémentaires
Continuer la documentation
Meeting de groupe
Création de la classe qui contient la logique de l'application
Création de la classe pour ouvrir le répertoire
Création de la classe pour lire les fichiers Excel
Création de la classe pour lire les fichiers PDF
Création de la classe pour lire les fichiers txt et autres (fichiers de programmations -> php, js, ...)
Création de la classe pour lire dans les fichiers Word
Création de la classe qui trie par auteurs
Création de la classe qui trie par dates
Création de la classe qui trie par nom de fichiers
Correction des éventuels bugs

2.4.5 Sprint 5


Sprint 5 - Correction de bug et améliorations
Affichage des documents plus agréable pour l'utilisateur
Exécution de la recherche à l'aide de la touche "entrée"
Modification de l'interface
Restructuration du code pour améliorer la vitesse d'exécution
Meeting de groupe
Recherche pour lire du Excel avec Spire
Gérer les formats de fichiers à ne pas trier
Recherche sur le déploiement
Test du déploiement
Factorisation du code

2.5 Dossier de conception

Nous avons décidé de créer un répertoire sur GitHub, ce qui nous permet d'avoir toujours nos fichiers à jour entre les deux membres du groupe.

Logiciels utilisés :

- GitHub Desktop, pour pouvoir synchroniser nos fichiers sur le serveur de GitHub.
- La suite Office, pour créer les fichiers nécessaires pour le projet.
- Navigateur web par défaut, pour aller sur internet.
- Visual Studio 2017, pour créer l'application pour les clients.
- Ordinateur sous Windows 10, ordinateur mis à disposition par le CPNV.
- Balsamiq, création des maquettes de l'application.
- Trello, pour la planification du projet.

2.6 Use cases / Scénarios

Mon application sert à :

- Trier des fichiers
- Retrouver des fichiers
- Interagir avec des fichiers

Se référer au point 2.8 Interface finale

2.6.1 Trier des fichiers

2.6.1.1 Aucun critère sélectionné méthode 1

Action	Situation particulière	Réaction
L'utilisateur clique sur « Sélectionner »		Une fenêtre d'explorateur Windows s'ouvre
L'utilisateur sélectionne le dossier dans lequel il veut faire ses recherches		Les différents documents se trouvant dans le dossier s'affichent

2.6.1.2 Aucun critère sélectionné méthode 2 + erreurs

Action	Situation particulière	Réaction
L'utilisateur clique sur le bouton « rechercher »	Aucun dossier n'est sélectionné	Un message d'erreur indique « Aucun dossier sélectionné. »
L'utilisateur écrit manuellement le chemin du dossier	Le dossier n'existe pas	Un message d'erreur indique « Le dossier sélectionné n'existe pas. »
L'utilisateur écrit correctement le chemin du dossier		Les différents documents se trouvant dans le dossier s'affichent

2.6.1.3 Dossier vide

Action	Situation particulière	Réaction
L'utilisateur clique sur « Sélectionner »		Une fenêtre d'explorateur Windows s'ouvre
L'utilisateur sélectionne le dossier dans lequel il veut faire ses recherches	Aucun fichier n'est retrouvé dans ce dossier	Un message d'erreur indique « Il n'existe aucun fichier dans le dossier sélectionné. »

2.6.2 Retrouver des fichiers

2.6.2.1 Recherche d'un fichier spécifique + erreur

Action	Situation particulière	Réaction
L'utilisateur clique sur « Sélectionner »		Une fenêtre d'explorateur Windows s'ouvre
L'utilisateur sélectionne le dossier dans lequel il veut faire ses recherches		Les différents documents se trouvant dans le dossier s'affichent
L'utilisateur clique dans le champ « auteur » ou/et « mots-clés » ou/et « date de modifications » ou/et « nom du fichier » et entre une/des valeur(s) à l'intérieur		
L'utilisateur clique sur « Rechercher »	Aucun fichier n'est trouvé	Un message d'erreur indique « Aucun fichier ne correspond aux critères de recherche. »
L'utilisateur change la/les valeur(s) du/des champ(s) qu'il a rempli puis clique sur « Rechercher »		Les différents documents qui se trouvent dans le dossier et correspondent aux critères s'affichent

2.6.3 Interagir avec des fichiers

2.6.3.1 Ouvrir le fichier + erreur

Action	Situation particulière	Réaction
	L'utilisateur a déjà trouvé le fichier qu'il recherchait	
L'utilisateur clique sur « Ouvrir le fichier »	Aucun fichier n'est sélectionné	Un message d'erreur indique « Aucun fichier n'a été sélectionné. »
L'utilisateur fait un clic gauche sur le nom du fichier voulu		Le nom du fichier est surligné en bleu
L'utilisateur clique sur « Ouvrir le fichier »		Le fichier s'ouvre avec le logiciel utilisé par défaut

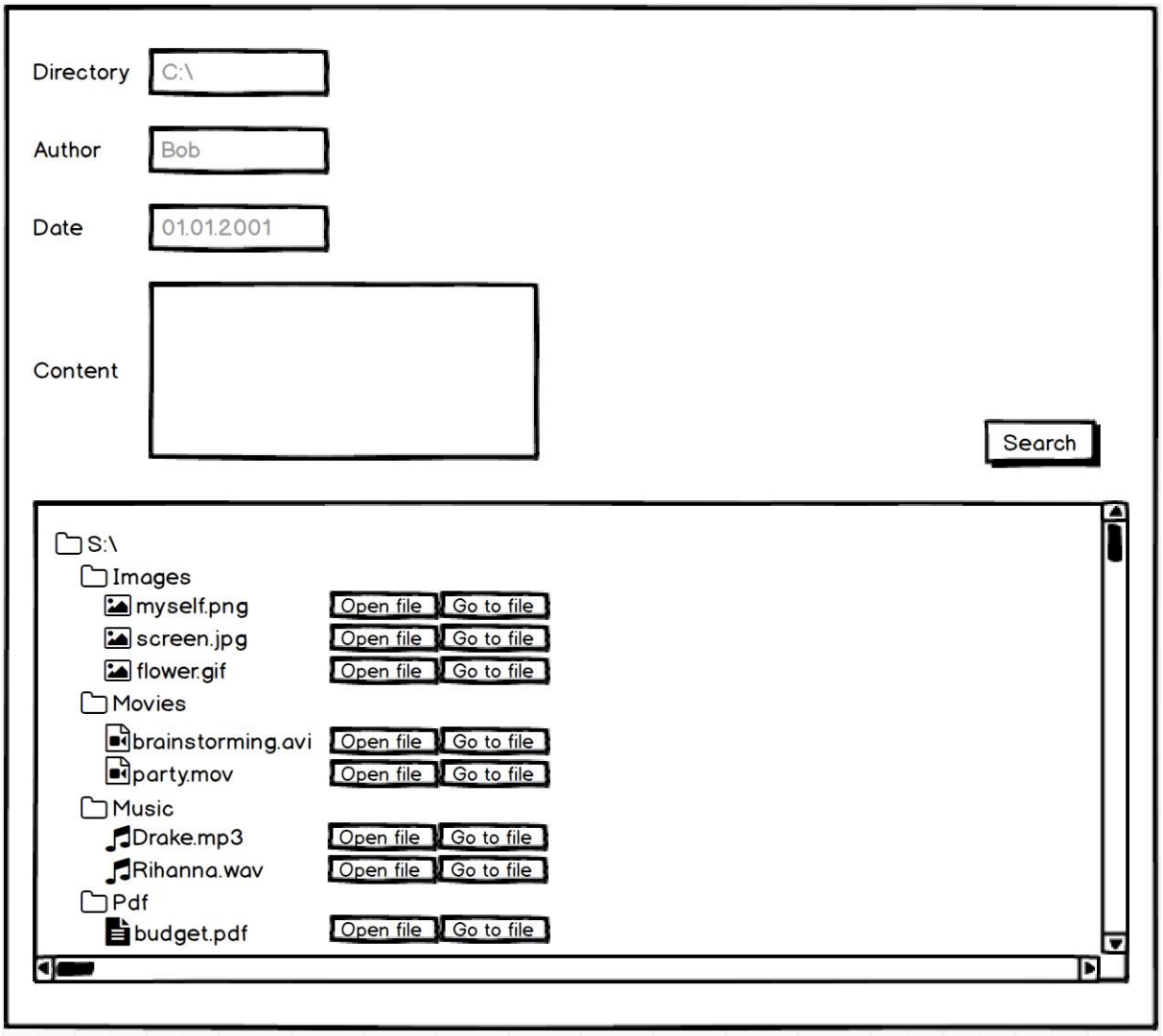
2.6.3.2 Se rendre au lieu réel du fichier dans l'explorateur Windows + erreur

Action	Situation particulière	Réaction
L'utilisateur clique sur « Ouvrir le répertoire »	Aucun répertoire n'a été sélectionné	Un message d'erreur indique « Aucun répertoire n'a été sélectionné. »
L'utilisateur sélectionne un dossier		
L'utilisateur clique sur « Ouvrir le répertoire »	Aucun fichier n'est sélectionné	L'explorateur Windows s'ouvre est affiche le contenu du répertoire
L'utilisateur ferme l'explorateur Windows, clique sur un fichier puis clique sur « Ouvrir le répertoire »	Un fichier est sélectionné	L'explorateur Windows s'ouvre et un focus sur le fichier sélectionné est fait

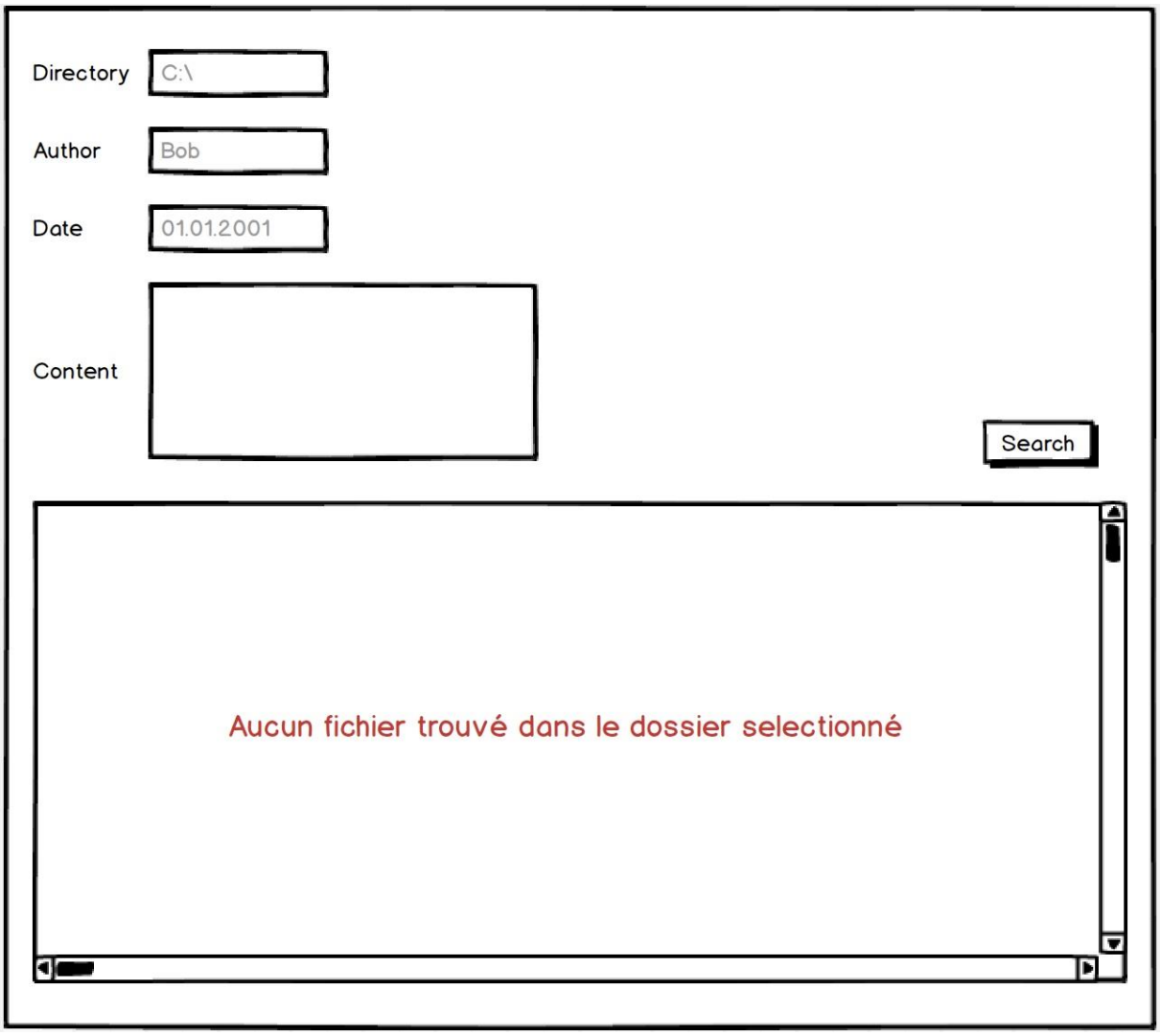
2.6.3.3 Fermer la fenêtre

Action	Situation particulière	Réaction
	L'utilisateur a déjà trouvé le fichier qu'il recherchait	
L'utilisateur fait un clic gauche sur le fichier voulu		Le fichier est sélectionné en bleu
Il clique sur le bouton « Ouvrir le répertoire »		La fenêtre du répertoire s'ouvre
L'utilisateur clique sur la croix rouge de la nouvelle fenêtre		La fenêtre de l'explorateur Windows se ferme

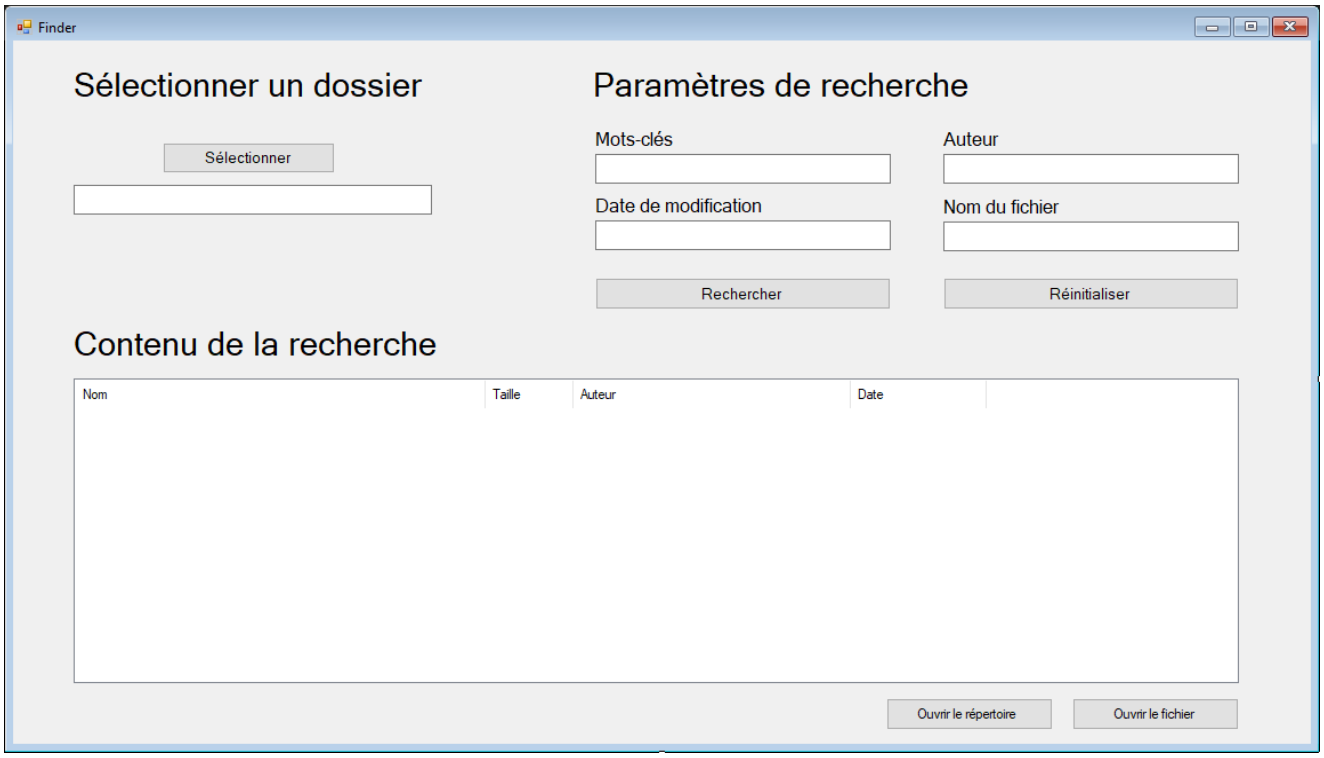
2.7 Maquettes



Voici la page qui s'affiche après avoir lancé la recherche dans un dossier spécifique, en indiquant l'auteur des fichiers et une date de création.


Directory
C:\
Author
Bob
Date
01.01.2001
Content
Search
Aucun fichier trouvé dans le dossier selectionné

2.8 Interface finale



3 Réalisation

3.1 Dossier de réalisation

En ce qui concerne le dossier de réalisation, il peut être mis n'importe où sur un ordinateur. Étant donné que notre projet est sauvegardé sur GitHub, on doit cloner le répertoire complet en local sur l'ordinateur pour pouvoir avoir accès au projet.

Une fois le clone effectué (procédure de déploiement expliquée plus bas), le répertoire contient :

Le dossier documentation contient :

- La documentation du projet, les images, le cahier des charges pour les clients.
- Aussi la maquette du site, que nous avons réalisé.

Le dossier finder contient :

- Toute notre application que nous avons développé avec Visual Studio.
- Des autres sous-dossiers qui contiennent les packages que nous avons utilisés, les classes et l'exécutable de l'application.

Le dossier journal de travail contient :

- Le journal de travail de chaque personne du groupe.

3.2 Description des tests effectués

Nous avons créé un répertoire dans lequel se trouve des fichiers (environ 500) de tous types (pdf, xlsx, docx, doc, pptx, png, php, js, html, log, etc…), ils contiennent tous du texte et certains sont plus volumineux que d'autres. Les tests ont été réalisés en suivant les use cases, même lorsque la fonctionnalité était terminée, nous avons continué de vérifier de temps à autre si elle était toujours opérationnelle ou s'il y avait un conflit lors de l'implémentation ou d'une factorisation du code. Des tests ont également été effectués sur d'autre PC et nous avons fait tester notre application à des camarades de classe.

Lorsque nous sommes arrivés en fin de projet, nous avons créé 14'000 fichiers afin de tester la robustesse et la rapidité de notre application.

3.3 Erreurs restantes

Il n'a a priori pas d'erreurs restantes, mais nous pouvons énumérer quelques améliorations à entreprendre. Notamment le fait de pouvoir lire le contenu d'un fichier PowerPoint lorsque nous mettons un mot-clé dans le tri.

Une autre amélioration serait d'améliorer la rapidité lorsqu'il y a 14'000 fichiers, en implémentant linQ, par exemple.

3.4 Liste des documents fournis

Vous trouverez ci-dessous la liste des fichiers fournis :

- Journal de travail de chaque personne du groupe
- La documentation du projet
- Le cahier des charges (déjà fourni)
- Le dossier complet de l'application en plus du fichier exécutable
- Le lien vers notre répertoire sur GitHub (qui sera envoyé par mail)

4 Conclusions

Pour conclure cette documentation ainsi que le projet, voici quelques lignes. Nous avons décidé de faire notre application avec l'aide de Visual Studio. Le langage de programmation de ce programme nous a demandé quelques temps d'adaptations pour apprendre la mécanique et créer l'application. Toutes les fonctions que les clients demandaient ont été implémentées. La lecture dans les fichiers PowerPoint ne se fait pas, nous n'avons pas trouvé une solution pour la lecture dans ce type de fichier.

Le projet en lui-même s'est bien déroulé, nous avons partagé le travail en fonction des facilités de chacun. Nous avons quand même développé l'application ensemble, quand nous avions des soucis nous recherchions des solutions sur internet et entre nous.

4.1 Conclusion Alexandre

Lors de ce projet j'ai eu l'occasion de revoir le C# car pendant mes années de CFC j'ai beaucoup plus développer en PHP/JS. Je pense que ce projet m'a permis de revoir le C# d'un autre point de vue et de me rendre réellement compte de ce qu'on peut faire avec, pendant mes recherches j'ai également vu plein de possibilités qui étaient offertes par le C#.

Concernant l'utilisation de GitHub en groupe, je l'ai trouvée un peu confuse et j'ai eu l'occasion d'apprendre à m'en servir d'une manière plus efficace lors d'autres cours, je n'ai donc pas pu utiliser GitHub correctement lors de ce projet mais nous avons tout de même eu le résultat attendu.

4.2 Conclusion Jérémy

Le projet m'a permis de refaire du C# ce qui n'était plus arrivé depuis la deuxième année lorsque j'étais en CFC. Il m'a permis de découvrir des fonctionnalités que je ne connais pas sur Visual Studio, comme le fait de pouvoir générer un setup.exe pour installer notre application sur une machine, ou encore le package Spire qui permet d'extraire des données dans des fichiers de tous types.

Il a fallu penser aussi à remplir et tenir un journal de travail ce qui n'est pas toujours évident lorsque l'on arrive en fin de période ou qu'on est à fond dans le projet pour vite finir une fonctionnalités ou autres.

5 Annexes

5.1 Sources – Bibliographie

Nous avons regardé comment faire certaines de nos fonctions sur le site de Spire :

- https://www.e-iceblue.com/Tutorials.html

5.2 Journal de travail

Se référencer aux fichiers de, Jérémy et Alexandre, mis en annexe.

5.3 Manuel d'installation

Cloner notre répertoire sur GitHub (https://github.com/JeremyGfeller/MAW_1.1). Une fois sur le site, vous serez dirigé sur la page d'accueil de notre projet. Pour cloner notre répertoire, cliquez sur le gros bouton « clone or download » en haut à droite. Ensuite, cliquez sur « Download ZIP ». Une fois le dossier télécharger, rendez-vous dans le dossier où il s'est téléchargé.

Dès que le dossier est trouvé, faites un clic droit sur le dossier télécharger, et cliquez sur « Extraire ici ». Rendez-vous ensuite dans le dossier extrait.

Une fois dans le dossier télécharger, pour lancer l'application, lancer le fichier qui s'appelle « Finder.exe ».

Rien d'autres ne doit être installé.

5.4 Manuel d'utilisation

Pour ce chapitre je vais vous expliquer comment utiliser notre application.

Tout d'abord, elle se compose de la manière suivante, dans le coin en haut à gauche, vous trouverez un bouton pour sélectionner le dossier où s'effectuera la recherche. Une fois le dossier sélectionné, le contenu de celui-ci s'affichera dans le grand encadré en bas au milieu.

Dans le coin en haut à droite se trouve les champs de recherches. Le champs mot-clé va aller vérifier dans chaque fichier si le mot que nous recherchons se trouve dans un fichier. S'il y a un correspondance le contenu s'affiche dans l'encadré du milieu.

Le champ auteur va regarder l'auteur du fichier avec celui que l'utilisateur aura entré dans le champ. S'il y a une correspondance il affichera les fichiers qui auront l'auteur recherché.

Quand on rentre une date ou simplement un chiffre du jour, du mois ou de l'année il va contrôler dans les fichiers s'il y a une correspondance.

En entrant le nom d'un fichier dans le dernier champ, il va regarder dans le dossier si un fichier correspond à ce que l'utilisateur à entrer.