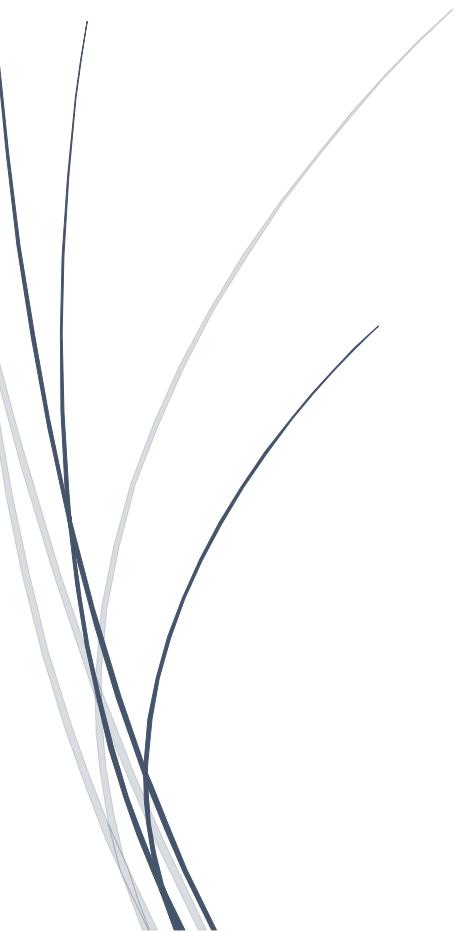


08/06/2020

Documentation Scrum'o'Wall

Travail de Diplôme Technicien ES en
informatique.

Session 2019 2020



Travail de Gaël Mariot
Suivi par Anne Terrier
CENTRE DE FORMATION PROFESSIONNEL TECHNIQUE

1	Résumé & Abstract	2
1.1	Résumé	2
1.2	Abstract.....	2
2	Lexique	3
3	Introduction	4
4	Cahier des charges	4
4.1	Objectifs du projet.....	4
4.2	Fonctionnalités.....	4
4.3	SWOT	4
4.4	Étude de l'existant.....	5
4.5	Planning	8
4.6	Outils.....	9
5	Analyse fonctionnelle.....	10
5.1	Description des fonctionnalités.....	10
5.2	Maquettes d'écrans.....	11
5.3	Fonctionnalités additionnelles.....	22
5.4	Manuel d'installation.....	23
6	Analyse Organique	24
6.1	Modèle de données.....	24
6.2	Modèle de classes	26
6.3	Apports personnels	30
6.4	Communications avec la base de données	30
6.5	Liaison Access.....	31
6.6	Glisser-déposer multipoint	31
7	Plan de tests.....	34
7.1	Test d'interface	34
7.2	Tests unitaires	37
8	Conclusion	38
9	Table des figures	39
10	Annexes	39

1 Résumé & Abstract

1.1 Résumé

Dans le cadre de ma formation en tant que Technicien en informatique au CFPT, il m'est demandé de réaliser un travail de diplôme pour démontrer mes capacités.

Le but de ce projet est de créer une application de gestion de projets agiles destinée à fonctionner sur un mur tactile. Mon idée de départ est de proposer un backlog sur lequel les membres d'une équipe Scrum peuvent interagir facilement. Afin d'enrichir l'application, j'y ai intégré d'autres fonctionnalités de gestion de projets et de gestion de sprints. On peut y trouver un diagramme de Burndown, des commentaires sur les User Stories, des checklists ainsi qu'une assignation d'utilisateurs aux projets/User Story/Objet de checklist.

Pour réaliser ce travail, je me suis orienté sur le développement d'une application centralisée intégrant les fonctionnalités tactiles. Toutes les données sont, par conséquent, stockées en local dans une base de données Access. L'application programmée en C# intègre la spécification graphique de Microsoft, Windows Presentation Foundation (WPF) pour l'interface utilisateur.

Afin de convenir au plus grand nombre, je me suis efforcé de rendre mon application modulable. Il est par exemple possible de personnaliser les colonnes disponibles dans les sprints via l'édition d'un projet. Il est également possible de lier des fichiers à une User Story pour les retrouver plus facilement. De plus, cette application peut gérer plusieurs utilisateurs simultanément grâce à une fonctionnalité multipoint ce qui me semble être un atout considérable. Je pense donc que c'est l'outil idéal pour remplacer les backlog papier.

1.2 Abstract

As part of my training as a technician in computer science at the CFPT, I am asked to complete a diploma work to demonstrate my capacities.

The purpose of this project is to create an application to manage agile projects to run on a touch wall. My initial idea is to provide a backlog on which Scrum team members can easily interact. To enrich the application, I integrated other project management and sprint management features. You can find a Burndown chart, a comment section on User Stories, checklists on User Stories as well as a user assignment on Projects, User Story and Checklist items.

To achieve this work, I focused on the development of a centralized application integrating touch functionalities. All data is therefore stored locally in an Access database. The application made in C# integrates Microsoft's graphical specification, Windows Presentation Foundation (WPF) for the user interface.

To suit the largest number of people, I tried to make my application modular. For example, it is possible to customize the columns available in the sprints through the edition of a project. It is also possible to link files to a User Story to find them more easily. In addition, this application can manage several users simultaneously with a multitouch functionality which seems to me to be a considerable attribute. So, I think it is the ideal tool to replace paper backlogs.

2 Lexique

- **SWOT** : Une méthode d'analyse permettant de comparer un projet ainsi que sa faisabilité en analysant ses forces (Strength), ses faiblesses (Weaknesses), ses opportunités (Opportunities) et ses menaces (Threat).
- **Développement agile** : Le développement agile est une méthode de travail très répandue dans le pôle informatique car il permet de pouvoir maintenir un produit plus facilement et se concentre sur des petites parties d'un projet effectuées lors de sprint plutôt que sur le projet entier d'un coup.
- **Sprint** : Période de travail courte d'un projet.
- **User Story** : Courte description d'une fonctionnalité. Habituellement formulée en cas d'utilisation.
- **Backlog** : Ensemble des User Stories d'un projet.
- **MindMap** : Une méthode pour pouvoir récolter des idées sur un sujet ou une problématique.
- **CRUD** : Le CRUD est un acronyme signifiant « Create,Read,Update,Delete ». Il représente les opérations de création, lecture, mise à jour et suppression sur une base de données et sont les quatre opérations sont les piliers d'une base de données.
- **CRD** : Equivalent du CRUD mais sans la mise à jour.

3 Introduction

La validation de la formation de technicien ES en informatique du Centre de Formation Professionnel Technique nécessite la réalisation d'un projet sur une durée de neuf semaines afin de démontrer les compétences acquises durant ce cursus.

Ce projet porte sur la création d'un logiciel de gestion. En effet, l'informatique permet de faciliter la gestion de bien des façons et, ayant déjà accompli mon travail de CFC sur un outil de gestion, j'ai été intéressé par ce projet quand Mme. Terrier me l'a proposé.

4 Cahier des charges

4.1 Objectifs du projet

Le but de ce projet est de créer un logiciel interagissant avec le mur tactile NCI Lab situé dans la salle de Technicien ES du CFPT Informatique. Ce logiciel permettra la gestion de projets en méthode agile et utilisera la fonctionnalité multipoint de l'écran permettant ainsi l'utilisation du logiciel par plusieurs utilisateurs simultanément.

4.2 Fonctionnalités

4.2.1 Stockage multi-projet

Le logiciel devra pouvoir contenir plusieurs projets et les garder en mémoire afin de permettre à plusieurs équipes de travailler sur le même mur. Cela permet d'apporter un gain de place aussi bien sur le stockage de la machine que physiquement.

4.2.2 Historique des sprints

Le logiciel permettra de voir les sprints selon leur ordre chronologique pour permettre aux utilisateurs de voir le chemin qu'ils auront parcouru et ils pourront planifier en avances les sprints suivants.

4.2.3 Détection multipoint

Le logiciel pourra gérer une interaction multipoint permettant à plusieurs utilisateurs de travailler sur tout l'écran simultanément.

4.3 SWOT

4.3.1 Forces

Ce projet permettra d'apporter une gestion plus facile et centralisée des projets avec les méthodes agiles. De plus, le fait de pouvoir interagir avec les tableaux rien qu'en les touchant rendra l'application encore plus intuitive.

4.3.2 Faiblesses

Ce projet est dangereux. Beaucoup d'applications reproduisent déjà le même rôle. Cependant, j'ai rajouté des options inédites telles que le diagramme de progression avec le burndown chart.

4.3.3 Opportunités

C'est la première fois que je crée une application prévue pour un appareil tactile avec C# et j'ai donc eu l'opportunité de pouvoir m'atteler à un projet qui pourra être utilisé tout en découvrant la détection multipoint avec C#.

4.3.4 Menaces

Malheureusement, le mur tactile n'est plus mis à jour et la dernière version des pilotes du mur tactile est prévue pour Windows 8. De plus, le mur tactile est difficile à calibrer afin de ne pas avoir d'erreur de réception.

4.4 Étude de l'existant

Afin de pouvoir comparer, j'ai sélectionné des applications qui gèrent les fonctions tactiles. Je me suis arrêté aux plus connues.

4.4.1 Trello

Trello est un produit racheté par la compagnie « Atlassian ». Il permet d'afficher un tableau contenant diverses listes créées par l'utilisateur. Dans ces listes, des cartes sont créées et on peut leur attribuer des utilisateurs, des mots-clefs ou encore créer une checklist afin d'accomplir diverses tâches pour finir une carte.

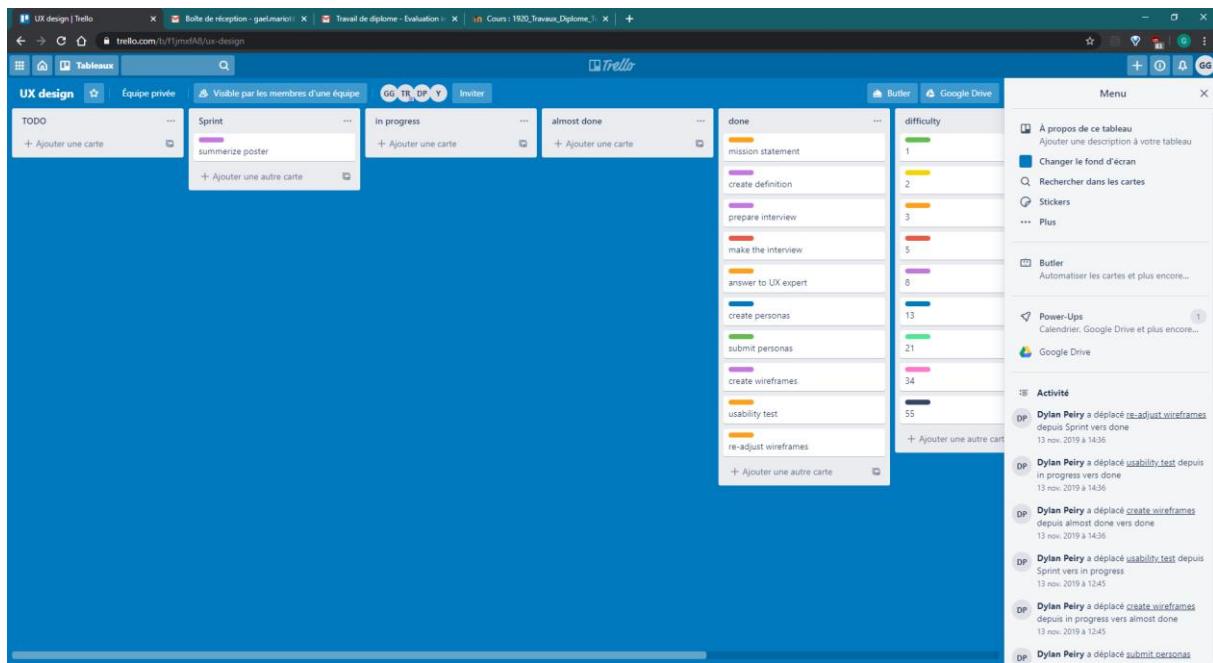


Figure 1 Capture d'écran de trello

<https://trello.com/>

4.4.2 JIRA

JIRA est un produit de la compagnie « Atlassian ». Il permet de créer et distribuer des tâches aux divers employés. Une fenêtre permet de définir des dates cible pour sortir certaines versions. Il existe un moyen de pouvoir visualiser les performances de son équipe en temps réel.

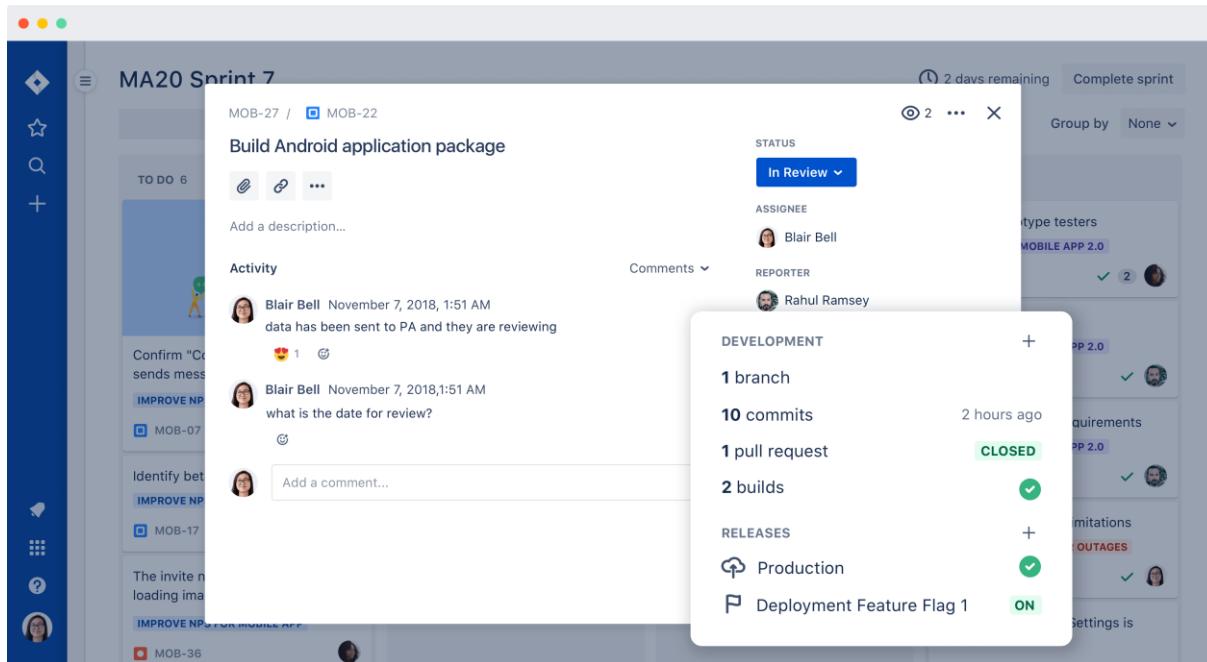


Figure 2 Capture d'écran de JIRA

<https://www.atlassian.com/software/jira>

4.4.3 Ubikey

Ubikey est une application payante. Elle permet de travailler à plusieurs sur le même écran et de connecter plusieurs écrans à distance. La collaboration via un téléphone portable ou un autre ordinateur est également possible.

Cependant, cette application ne propose pas d'outils de visualisation de la progression.

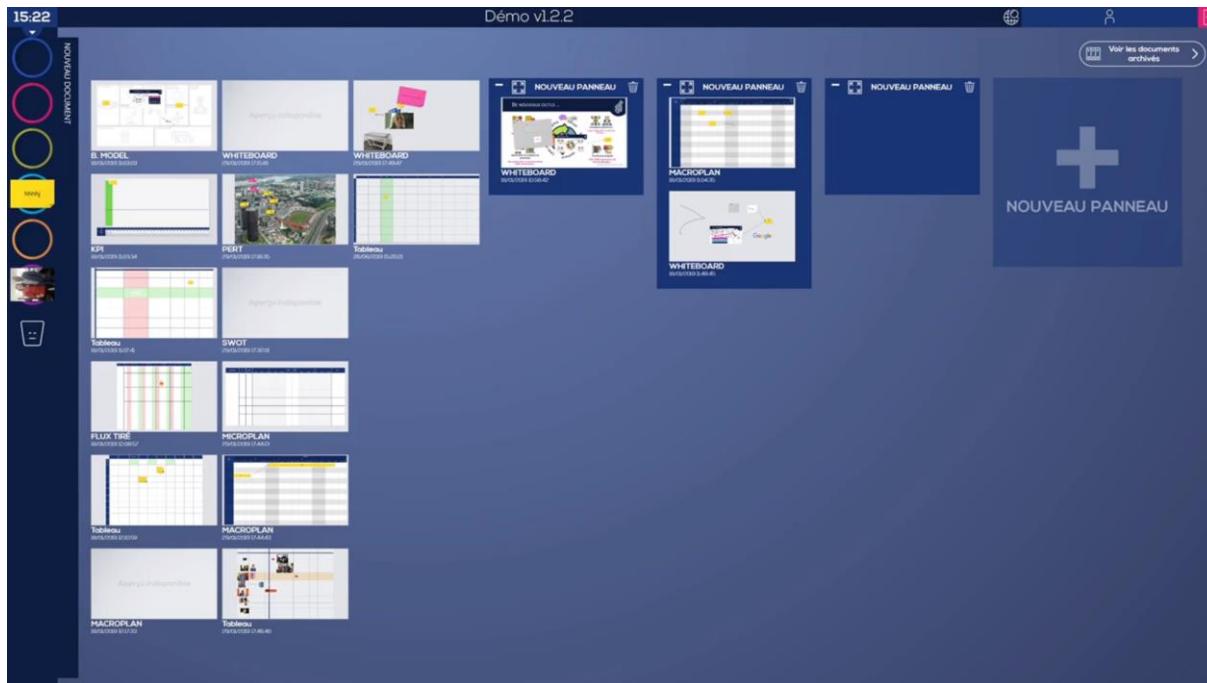


Figure 3 Capture d'écran de Ubikey

<https://www.ubikey.fr/>

4.4.4 Kantee

Kantee est un logiciel de gestion de projet payant. Il permet de collaborer en temps réel sur plusieurs dispositifs et d'afficher les Sprints sous diverses formes telles que le format classique en colonnes, une matrice contenant chaque colonne pour chaque employé ou encore un calendrier montrant le temps estimé. Il permet également de sauvegarder le projet en local pour continuer à gérer son projet en étant hors ligne.

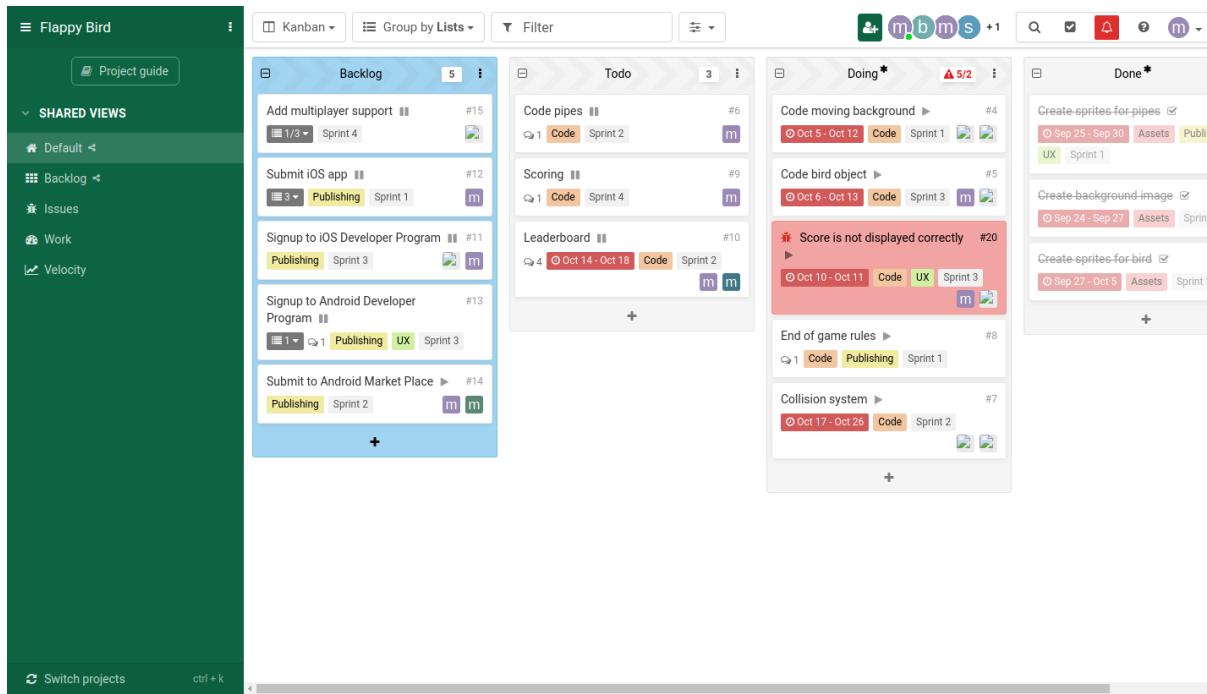


Figure 4 Capture d'écran de Kantee

<https://kantee.io/fr>

4.4.5 Résumé comparatif

Mon projet se distinguera de ces projets car toutes les données sont stockées localement et n'ont pas besoin d'internet. Cela signifie qu'aucun service ne doit être entretenu pour garder les données sur un serveur distant.

De plus, aucun de ces projets n'a de mindmap intégré dans leurs fonctionnalités. Le mindmap est une étape qui permet à l'équipe de développement de trouver un terrain d'entente plus facilement et de lier leurs pensées pour un même projet.

4.5 Planning

4.5.1 Prévu

Planning du travail de diplôme				
N° Tache	Description de la tâche	Temps estimé(j)	Temps réel (j)	Date limite
1	Documentation du projet, Journal de bord	18		
2	Modelisation de la base de données	1		
3	Création de la base de données	0.5		
4	Mise en place des systèmes de versioning	1		
5	Création du backend du logiciel	3		
6	Création de la fenêtre principale	1		
7	Création de la fenêtre de projet	1		
8	Création de la fenêtre de sprint	1		
9	Création d'un contrôleur	3		
10	Liaison contrôleur - fenêtre principale	1		
11	Liaison contrôleur - fenêtre de projet	1		
12	Liaison contrôleur - fenêtre de sprint	1		
13	Création du poster	2		
14	Implementation des mindmap	2		
15	Implémentation du gantt	2		
Jalon	Evaluation Intermédiaire 01			24.04.2020
Jalon	Rendu du rapport intermédiaire + poster			07.05.2020
Jalon	Evaluation Intermédiaire 02			08.05.2020
Jalon	Rendu version intermédiaire du résumé et de l'abstract			15.05.2020
Jalon	Evaluation Intermédiaire 03			22.05.2020
Jalon	Rendu travail			08.06.2020

Figure 5 Planning initial

4.5.2 Final

Planning du travail de diplôme				
N° Tache	Description de la tâche	Temps estimé(j)	Temps réel (j)	Date limite
1	Documentation du projet, Journal de bord	18	14	
2	Modelisation de la base de données	1	2	
3	Création de la base de données	0.5	1	
4	Mise en place des systèmes de versioning	1	0.25	
5	Création du backend du logiciel	3	2	
6	Création de la fenêtre principale	1	1	
7	Création de la fenêtre de projet	1	1	
8	Création de la fenêtre de sprint	1	1	
9	Création d'un contrôleur	3	2	
10	Liaison contrôleur - fenêtre principale	1	1	
11	Liaison contrôleur - fenêtre de projet	1	1	
12	Liaison contrôleur - fenêtre de sprint	1	1	
13	Création du poster	2	2	
14	Implémentation des mindmap	2	3	
15	Implémentation du gantt	2	0	
16	Création de fenêtres pop-up		3	
17	Tests unitaires		3	
Jalon	Evaluation Intermédiaire 01			24.04.2020
Jalon	Rendu du rapport intermédiaire + poster			07.05.2020
Jalon	Evaluation Intermédiaire 02			08.05.2020
Jalon	Rendu version intermédiaire du résumé et de l'abstract			15.05.2020
Jalon	Evaluation Intermédiaire 03			22.05.2020
Jalon	Rendu travail			08.06.2020

Figure 6 Planning final

On peut voir que certaines tâches ont fait leur apparition et que je n'ai pas eu tout le temps voulu afin de travailler sur la documentation.

4.6 Outils

4.6.1 Interface graphique

Afin de réaliser l'interface graphique, j'ai décidé d'utiliser WPF. En effet, la détection multipoint est déjà gérée sur WPF et non sur Windows Form. De plus, WPF permet de créer des applications multi-plateformes.

4.6.2 Base de données

Afin d'intégrer une base de données, j'ai choisi d'utiliser Access car c'est une technologie avec laquelle je suis déjà familier. En effet, n'étant déjà pas familier avec WPF et la détection multipoint, j'ai pensé qu'il serait mieux de ne pas miser sur trop d'inconnus et utiliser un système de base données que je connais bien et qui est très bien intégré avec C#. De plus, Access est bien plus facile à configurer pour l'utilisateur et ne requiert pas de serveur afin de le faire fonctionner.

4.6.3 Environnement

Afin de réaliser ce projet, j'ai à ma disposition un ordinateur avec Windows 10 ainsi que l'édition Community de *Visual Studio 2019*. On m'a également fourni un écran tactile *Iiyama ProLite T2735MSC* afin de pallier aux restrictions liées au Covid-19

Pour ce qui est des créations graphiques, je vais utiliser l'application graphique *GIMP* version 2.10.18 pour le poster, *Balsamiq Wireframes* pour les maquettes de l'interface utilisateur et *Draw.io* pour les diagrammes et le MCD

5 Analyse fonctionnelle

5.1 Description des fonctionnalités

Gestion des projets

- Créer, Modifier et Supprimer un projet
- Assigner des utilisateurs à un projet
- Assigner des états à un projet

Gestion des sprints

- Créer, Modifier et Supprimer un sprint
- Assigner des User Stories à un sprint

Gestion des User Stories

- Créer, Modifier et Supprimer une user story
- Assigner des utilisateurs à une user story
- Assigner des fichiers à une user story
- Assigner des listes à une user story
- Commenter une user story
- Voir l'historique des actions d'une user story

Gestion des listes

- Créer, Modifier et Supprimer une liste
- Ajouter des objets à la liste
- Assigner des utilisateurs

Gestion des utilisateurs

- Créer, Modifier et Supprimer un utilisateur

Gestion des états

- Créer, Modifier et Supprimer un état

Gestion des fichiers

- Créer, Modifier et Supprimer un fichier

Autres

- Créer et Supprimer des commentaires
- Création automatique des activités

5.2 Maquettes d'écrans

5.2.1 Fenêtre principale

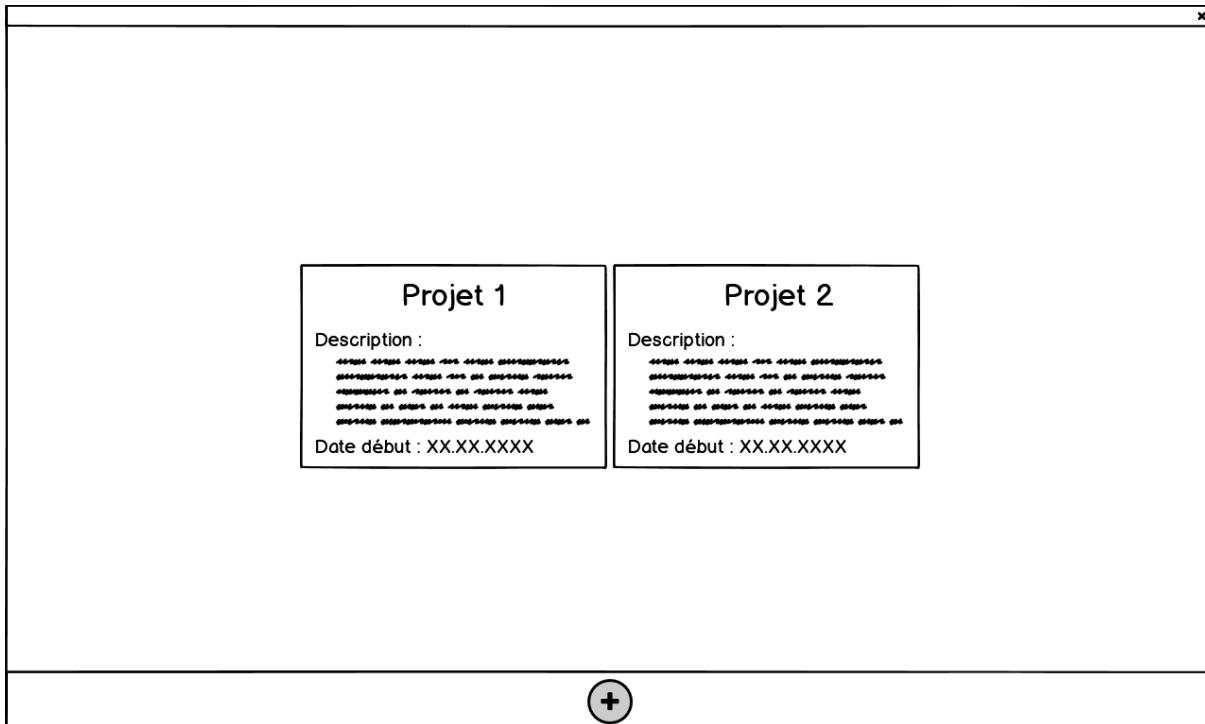


Figure 7 Maquette de la fenêtre de sélection de projet

Sur cette fenêtre, une mosaïque avec les différents projets déjà créé sera afficher. Un bouton permettra de rajouter un projet. Un maintien prolongé permettra également de rajouter un projet via un menu contextuel.

5.2.2 Fenêtre de projet

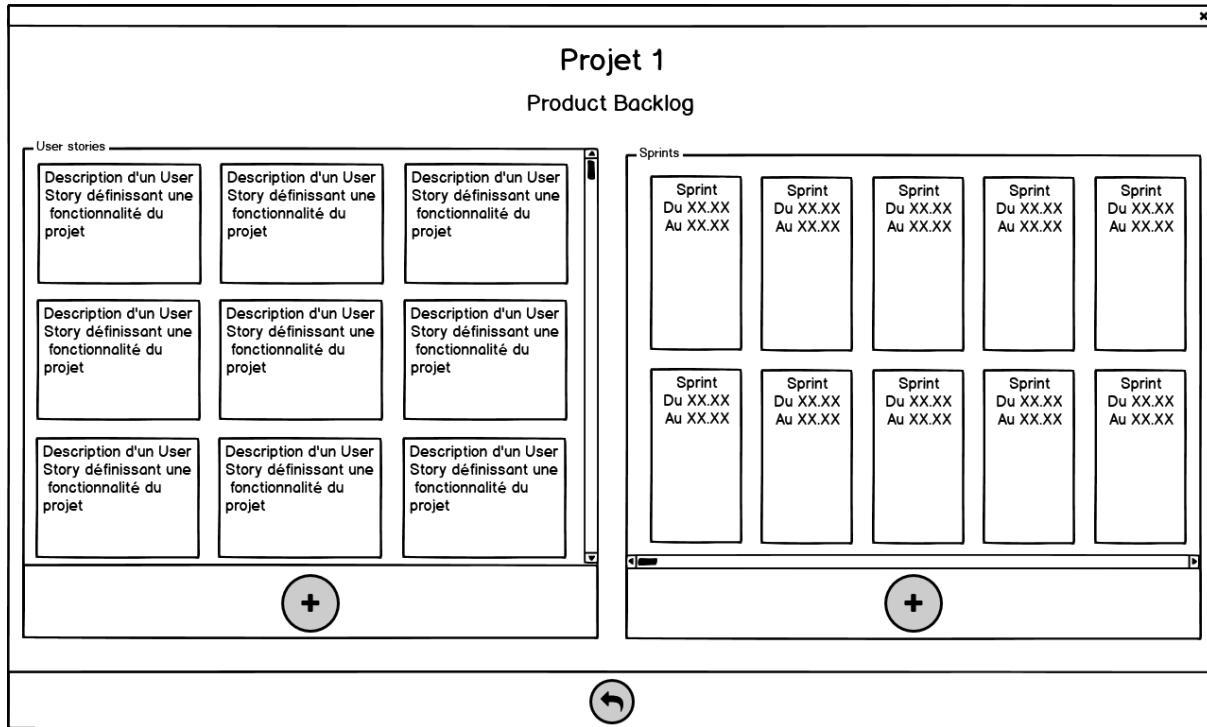


Figure 8 Maquette de la fenêtre de projet

Cette fenêtre s'affichera dès l'ouverture d'un projet. On peut changer le nom du projet ainsi que sa description en cliquant sur le bouton à côté du nom ou en déclenchant le menu contextuel via un appui prolongé.

La partie gauche permettra de créer des fonctionnalités à rajouter dans le projet à travers des Use Case. Elle permettra également de les modifier. La partie de droite, quant à elle, permettra de rajouter et d'éditer des Sprints et de leur rajouter des Use Case via un glisser-déposer sur le sprint voulu. Les sprints dont la date sera déjà passée seront légèrement grisés et ne pourront pas être modifiés. Le sprint en cours sera mis en évidence.

5.2.3 Fenêtre de sprint

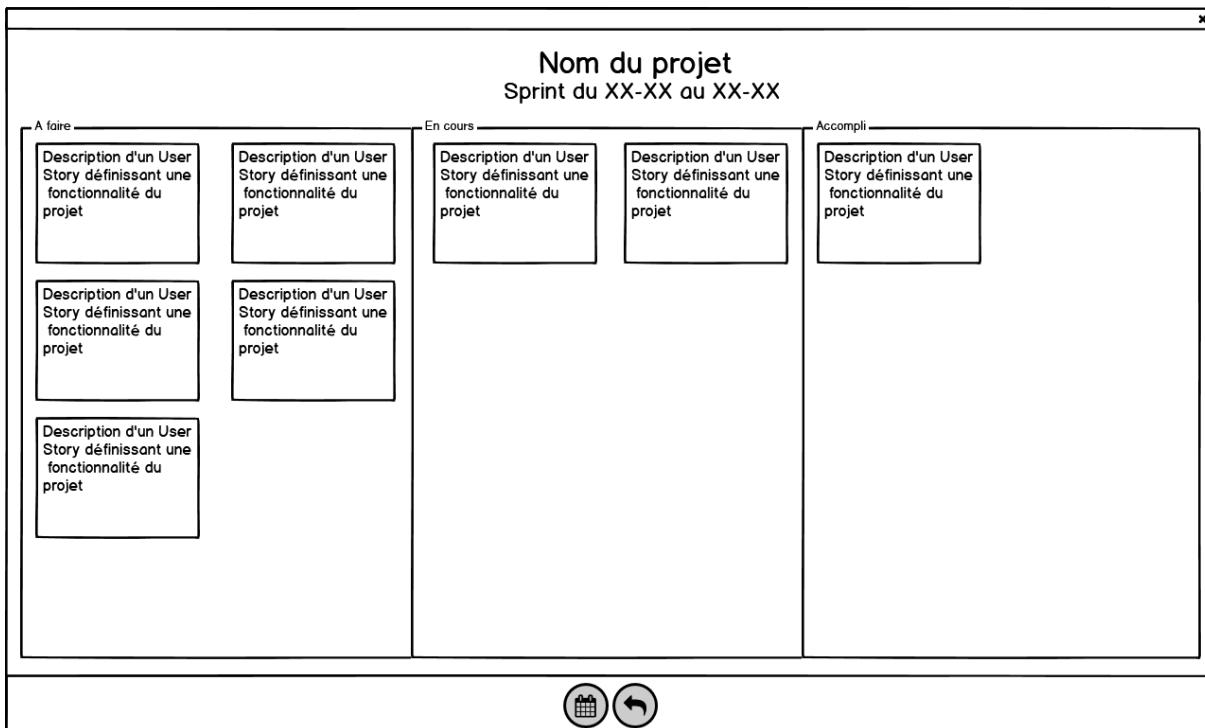


Figure 9 Maquette de la fenêtre de sprint

Cette fenêtre affiche le sprint sur lequel on aura appuyé. Un bouton permet de revenir au backlog. Un appui prolongé ouvrira le menu contextuel qui permettra de rajouter des Use Case ou de modifier les Use Case existants.

Le menu contextuel permettra de rajouter des colonnes pour permettre aux gens de gérer au mieux leur projet de la façon qui leur convient.

Le bouton avec le calendrier amène à la Fenêtre de burndown chart

5.2.4 Fenêtre de burndown chart

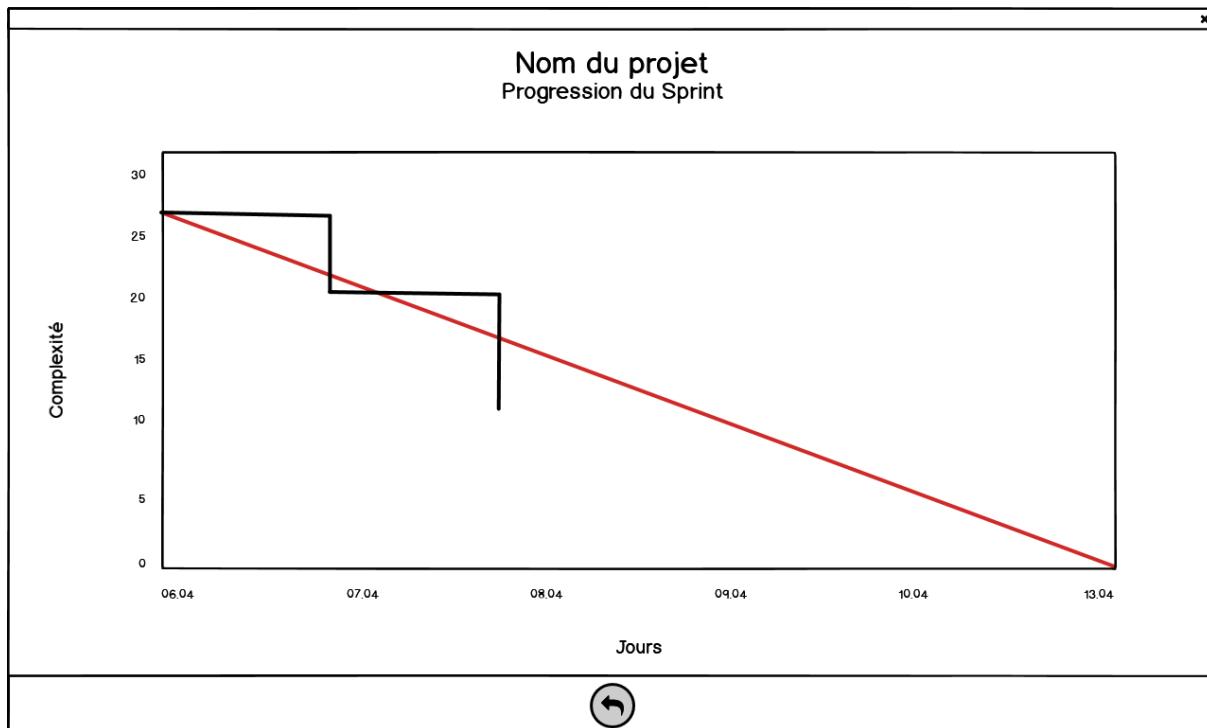


Figure 10 Maquette de la fenêtre du Burndown Chart

Cette fenêtre permet de visualiser la progression d'un sprint. La barre rouge représente l'avancée idéale du sprint et la barre noire représente l'avancée réelle du projet.

5.2.5 Fenêtre des utilisateurs

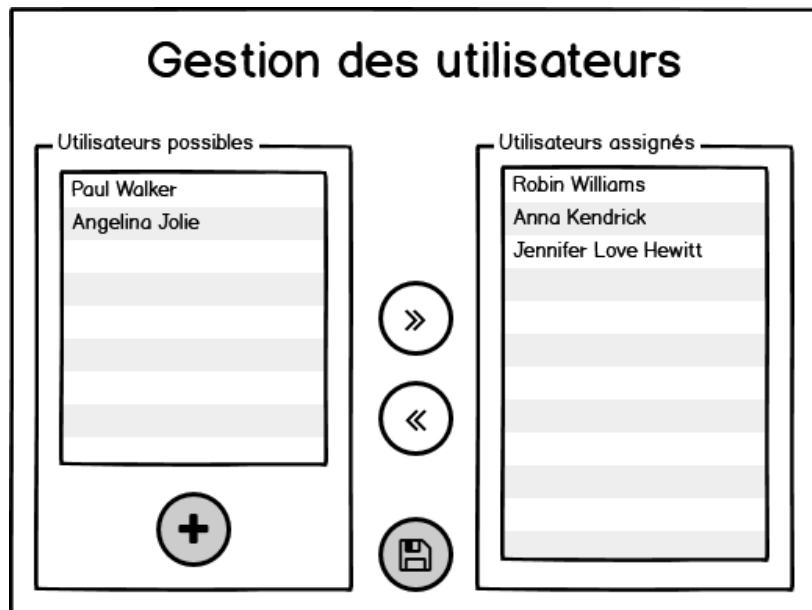


Figure 11 Maquette de la fenêtre des utilisateurs

Cette fenêtre permet de gérer les utilisateurs pour un UserStory, un projet ou un objet de checklist. Il affiche tous les utilisateurs et permet de rajouter et supprimer les utilisateurs assignés à l'objet. Cependant, aucun changement ne sera effectué si le bouton « Enregistrer » n'est pas pressé.

5.2.6 Fenêtre des états

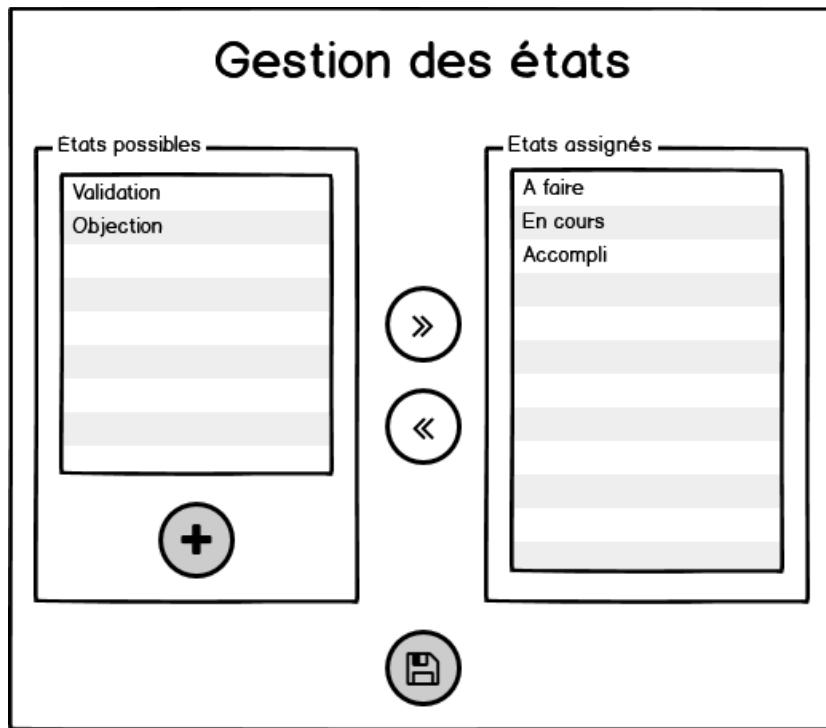


Figure 12 Maquette de la fenêtre des états

Cette fenêtre permet de gérer les utilisateurs pour un projet. Il affiche tous les états et permet de rajouter et supprimer les états au projet. Cependant, aucun changement ne sera effectué si le bouton « Enregistrer » n'est pas pressé.

5.2.7 Fenêtre des activités

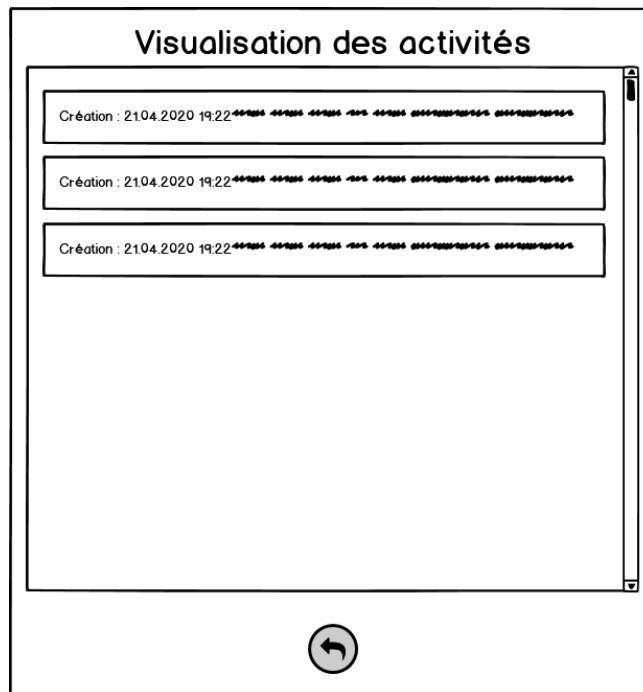


Figure 13 Maquette de la fenêtre des activités

Cette fenêtre permet d'afficher les activités concernant une User Story définie. Elle ne permet pas de créer d'activités car elles sont créées automatiquement.

5.2.8 Fenêtre des checklists

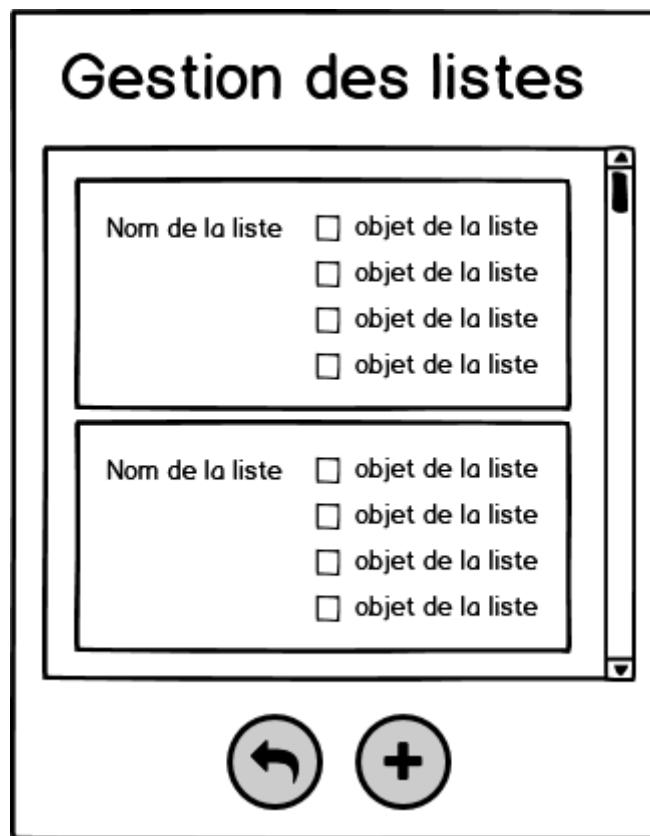


Figure 14 Maquette de la fenêtre des listes

Cette fenêtre permet de créer et de gérer les listes d'une User Story. En cliquant sur une liste, cela permet de modifier la liste. En appuyant sur le bouton « + », une redirection nous amène au pop-up de création de liste.

5.2.9 Fenêtre des commentaires

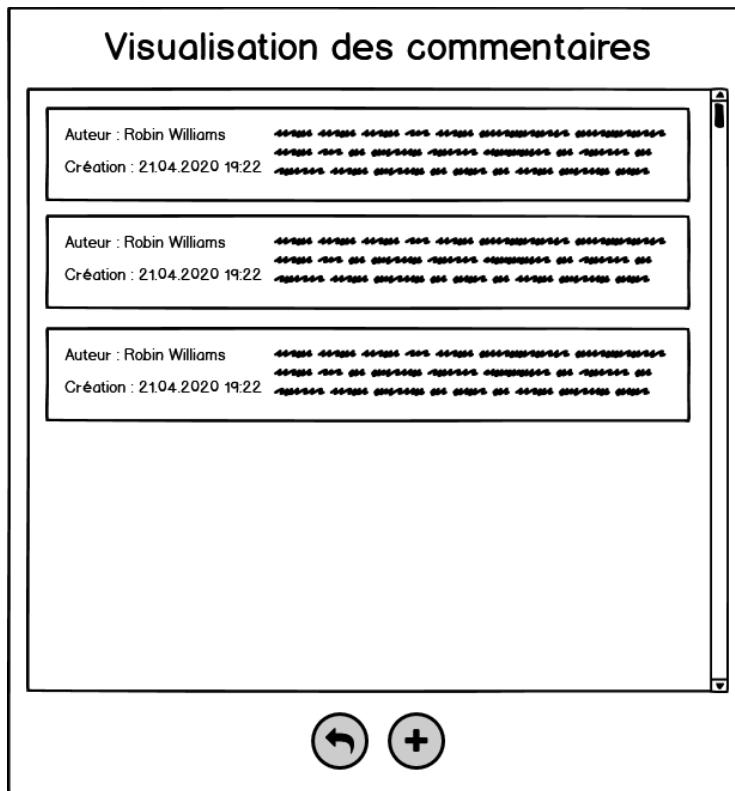


Figure 15 Maquette de la fenêtre des commentaires

Cette fenêtre permet de visualiser les commentaires. Les commentaires sont affichés par date et heure. Le bouton « + » permet de rajouter un commentaire en passant par le pop-up de création de commentaires.

5.2.10 Fenêtre des fichiers

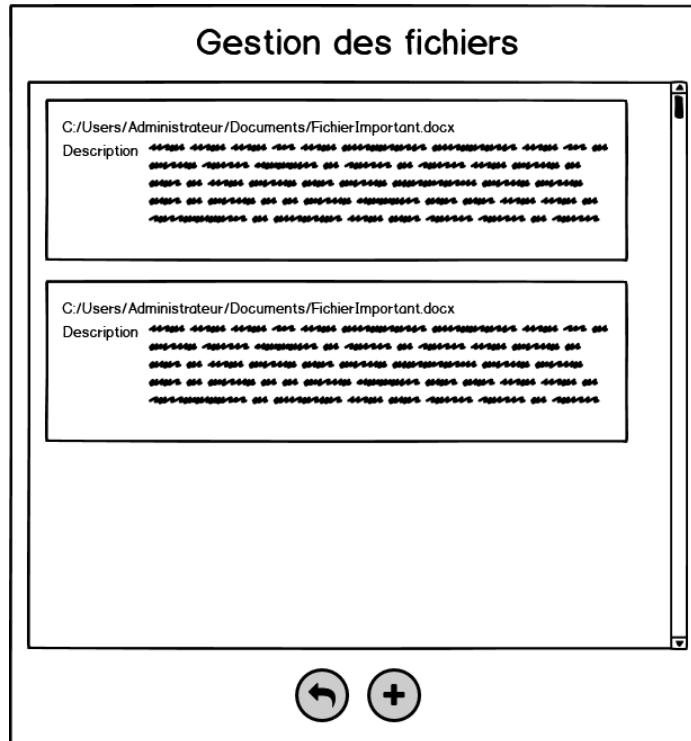


Figure 16 Maquette de la fenêtre de fichiers

Cette fenêtre permet de gérer les fichiers liés à une User Story. Appuyer sur un fichier permet de le modifier. Appuyer sur le bouton « + » permet d'ajouter un fichier.

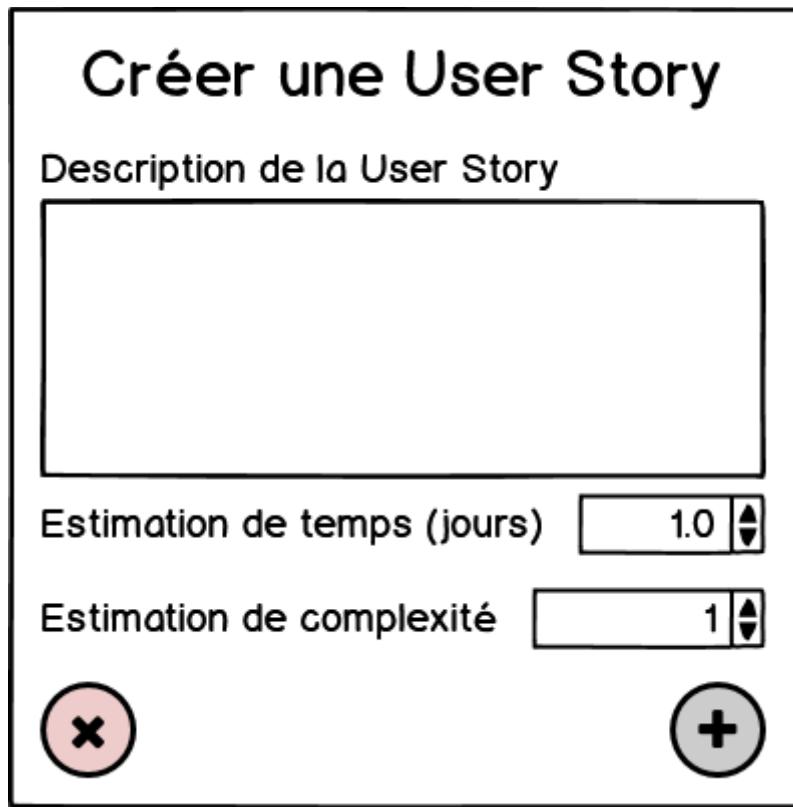
5.2.11 Pop-up Projets



Figure 17 Maquette du popup de création de projet

Ce popup apparaît quand la modification ou l'ajout d'un projet est demandé. En appuyant sur la croix en bas à gauche, la fenêtre se ferme sans changement alors qu'en appuyant sur la droite, les changements s'effectuent.

5.2.12 Pop-up UserStory



Créer une User Story

Description de la User Story

Estimation de temps (jours) 1.0

Estimation de complexité 1

x **+**

This form is used to create a User Story. It contains a large text area for the story description, two numeric input fields for estimation (in days and complexity), and two circular buttons with a minus sign and a plus sign respectively for adjusting these values.

Figure 18 Maquette du popup de création de UserStory

Ce popup apparaît quand la modification ou l'ajout d'une UserStory est demandée. En appuyant sur la croix en bas à gauche, la fenêtre se ferme sans changement alors qu'en appuyant sur la droite, les changements s'effectuent.

5.2.13 Pop-up Sprint



Créer un Sprint

Date de début aujourd'hui

Date de fin dans 7 jours

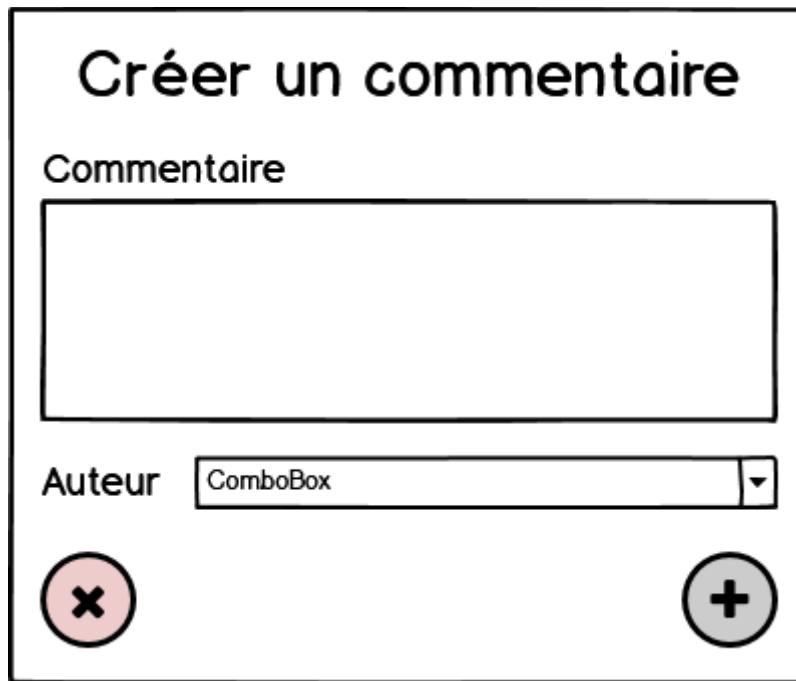
x **+**

This form is used to create a sprint. It includes date selection fields for the start and end dates, and two circular buttons with a minus sign and a plus sign for adjusting the duration.

Figure 19 Maquette du popup de création de sprint

Ce popup apparaît quand la modification ou l'ajout d'un sprint est demandé. En appuyant sur la croix en bas à gauche, la fenêtre se ferme sans changement alors qu'en appuyant sur la droite, les changements s'effectuent.

5.2.14 Pop-up des commentaires



Créer un commentaire

Commentaire

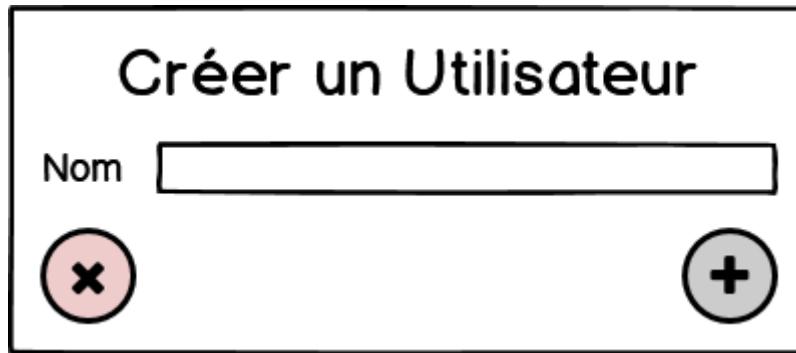
Auteur

X +

Figure 20 Maquette de pop-up de création de commentaire

Cette fenêtre permet de créer un commentaire. La date sera insérée automatiquement mais l'auteur devra être choisi parmi les utilisateurs assignés à la User Story. Le bouton « X » permet d'annuler la création. Le bouton « + » permet de confirmer la création si les champs requis ont été remplis.

5.2.15 Pop-up des utilisateurs



Créer un Utilisateur

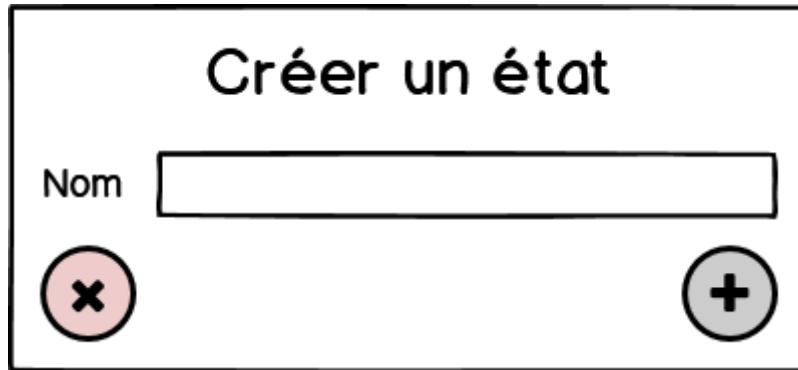
Nom

X +

Figure 21 Maquette de pop-up de création d'utilisateur

Cette fenêtre permet de créer un utilisateur en remplaçant le nom de l'utilisateur. Le bouton « X » permet d'annuler la création. Le bouton « + » permet de confirmer la création si les champs requis ont été remplis.

5.2.16 Pop-up des états



Créer un état

Nom

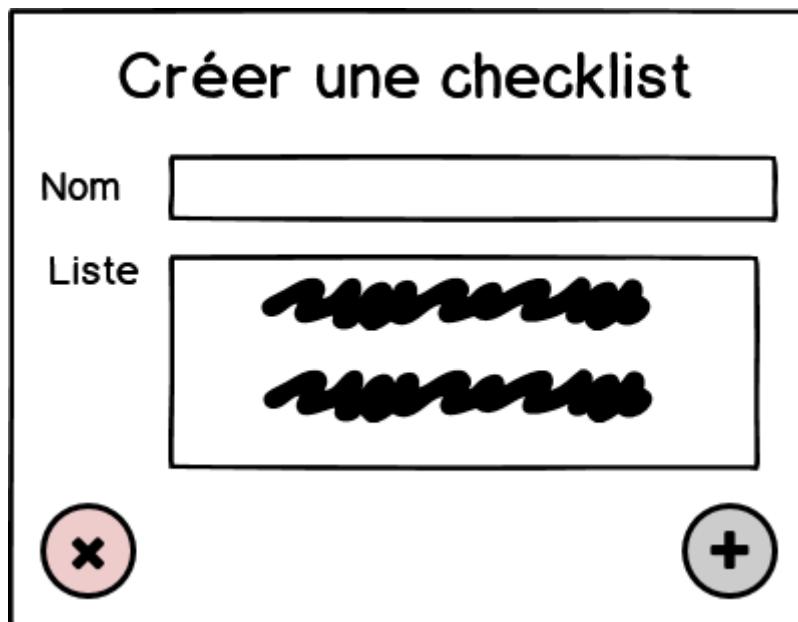
X +

A maquette de fenêtre intitulée "Créer un état". À l'intérieur, il y a un champ "Nom" avec une barre d'entrée vide. En bas à gauche, il y a un bouton circulaire avec une croix ("X"). En bas à droite, il y a un bouton circulaire avec un plus ("+").

Figure 22 Maquette de pop-up de création d'état

Cette fenêtre permet de remplir les informations nécessaires à la création d'un état. Il est nécessaire pour cela que le nom soit rempli. Le bouton « X » permet d'annuler la création. Le bouton « + » permet de confirmer la création si les champs requis ont été remplis.

5.2.17 Pop-up des checklists



Créer une checklist

Nom

Liste

X +

A maquette de fenêtre intitulée "Créer une checklist". À l'intérieur, il y a un champ "Nom" avec une barre d'entrée vide. En dessous, il y a un champ "Liste" contenant deux lignes de traits noirs. En bas à gauche, il y a un bouton circulaire avec une croix ("X"). En bas à droite, il y a un bouton circulaire avec un plus ("+").

Figure 23 Maquette de pop-up de création checklist

Cette fenêtre permet de remplir les informations nécessaires à la création d'une liste. Il est nécessaire pour cela que le nom soit rempli et que la liste ne soit pas vide. Le bouton « X » permet d'annuler la création. Le bouton « + » permet de confirmer la création si les champs requis ont été remplis.

5.2.18 Pop-up des fichiers

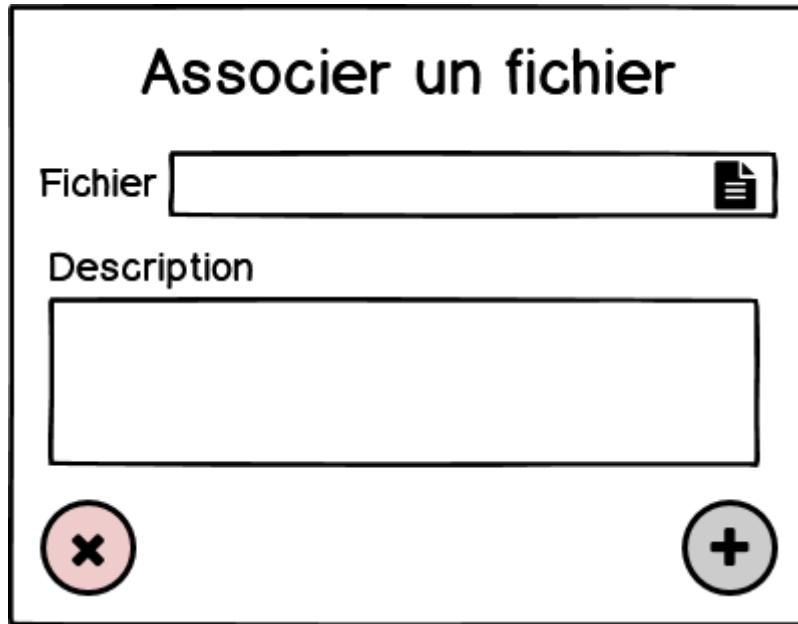


Figure 24 Maquette de pop-up de création fichier

Cette fenêtre permet d'associer un fichier local avec une User Story. Il est nécessaire de choisir un fichier et de donner une description. Le bouton « X » permet d'annuler la création. Le bouton « + » permet de confirmer la création si les champs requis ont été remplis.

5.3 Fonctionnalités additionnelles

Voici des fonctionnalités qui pourront être rajoutées si le temps le permet.

5.3.1 Gantt intégré

Le logiciel aura un diagramme de Gantt intégré pour chaque projet. Il se mettra à jour à chaque création de Use Case. Il permettra d'avoir une idée de la date de fin de projet et du nombre de jours nécessaires à la complétion d'un projet.

Fonctionnalités

- Affichage dans le temps

5.3.2 Mindmap intégré

Le logiciel pourra créer des mindmap. Cela permettrait aux équipes de pouvoir faire des brainstormings de façon plus efficace et sans avoir de problèmes pour retrouver le mindmap sur un aspect d'un projet.

Fonctionnalités

- Créer, Modifier et Supprimer un mindmap
- Ajouter, Modifier et Supprimer des nœuds de mindmap

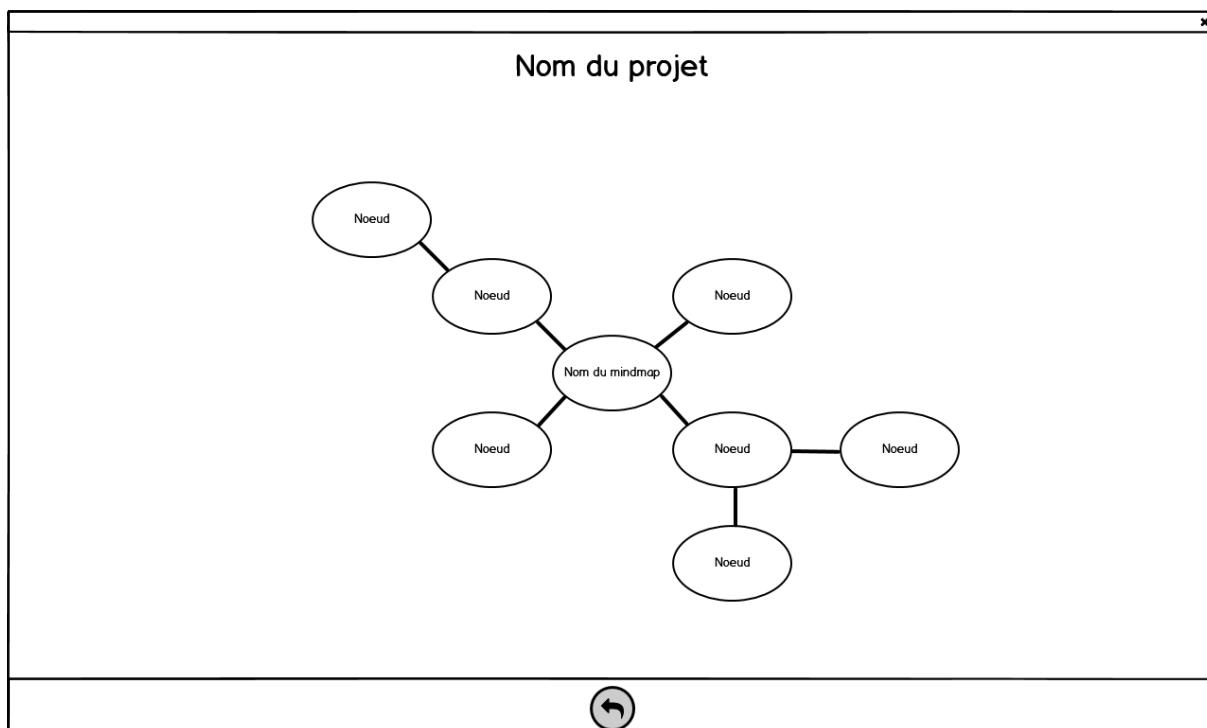


Figure 25 Maquette écran du mindmap

5.3.3 Protection de projets

Il pourrait être possible d'intégrer une gestion de mots de passe pour limiter l'accès aux projets aux autres utilisateurs du poste. Cela pourrait être utile si des informations sensibles se situent dans un projet.

Fonctionnalités

- Demande du mot de passe stocké dans la base de données

5.4 Manuel d'installation

5.4.1 Configuration de l'environnement

Afin de configurer l'environnement, il faut tout d'abord installer le moteur de base de données d'Access afin de pouvoir ouvrir les bases de données. Le moteur de base de données se situe sur cette page : <https://www.microsoft.com/fr-FR/download/details.aspx?id=39358>. De plus, le moteur de base de données étant créé uniquement en 64 bits, il n'est pas possible de lancer l'application en 32bits.

5.4.2 Activation du clavier virtuel

L'activation du mode tablette qui permet d'avoir le clavier virtuel quand un champ textuel est sélectionné n'est pas une possibilité. En effet, la présence de plus d'un écran empêche l'activation de cette fonctionnalité.

Cependant, il est possible d'activer cela sans activer le mode tablette. Cela requiert qu'aucun clavier ne soit connecté et qu'une option spécifique soit sélectionnée. Pour cela, il faut se rendre dans les **Paramètres > Périphériques > Saisie**. Dans ce menu, une option présente nécessite d'être activée. Elle est intitulée « **Afficher le clavier tactile lorsque vous n'êtes pas en mode tablette et qu'aucun clavier n'est connecté** »

5.4.3 Mise en place de la base de données

Afin d'avoir une base de données vide, trouvez le fichier « Empty.accdb » qui se situe dans le dossier « Installation ». C'est une base de données vierge contenant uniquement les données nécessaires.

Je vous conseille de copier ce fichier dans un endroit que vous retrouverez facilement et de le renommer avec un nom de votre choix.

6 Analyse Organique

6.1 Modèle de données

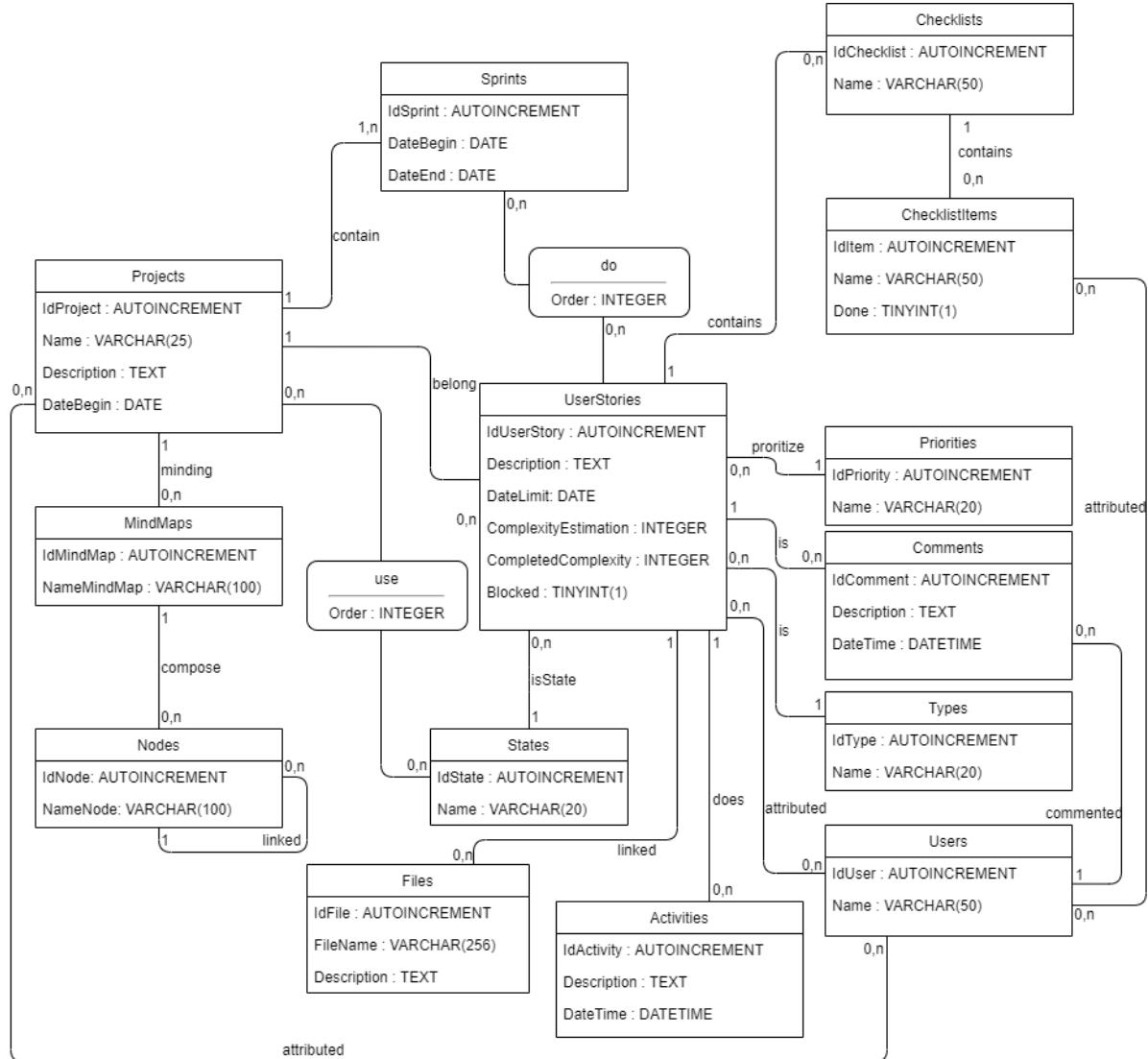


Figure 26 Modèle Conceptuel de Données

6.1.1 Table project

Cette table contiendra la description, le nom ainsi que la date de début du projet. C'est cela qui permettra d'identifier le projet sur la première page de l'application.

Cette table sera liée à la table *state* pour définir quelles colonnes seront affichés dans le sprint. Elle sera également liée à la table *user* pour déterminer l'attributions d'utilisateurs dans le projet.

6.1.2 Table UserStory

Cette table contient les différentes fonctionnalités. Chaque User Story contient une description de type TEXT, une estimation de complexité afin de pouvoir attribuer les tâches aux bonnes personnes, la valeur de la complexité accomplie pour pouvoir réaliser un suivi.

Une table de liaison reliera la table Sprint à cette table afin de créer le contenu du sprint. Une table de liaison avec tous les différents utilisateurs sera également créée pour montrer les utilisateurs attribués à cette tâche.

6.1.3 Table Sprint

Cette table permet de déterminer la date de début et de fin des sprints. On utilisera ces dates afin d'identifier les différents sprints et de les afficher dans la page du projet.

Une table de liaison reliera la table *UserStory* à cette table afin de créer le contenu du sprint.

6.1.4 Table State

Cette table stocke tous les états possibles qu'une tâche peut avoir.

Une table de liaison la reliera avec la table *project*. Cette table de liaison déterminera tous les états disponibles dans les sprints du projet.

6.1.5 Table MindMap

Cette table permet la création de mindmap en gardant leurs noms en mémoire et en les liant aux projets correspondants.

6.1.6 Table Node

Cette table contient tous les différents nœuds du mindmap. Cette table contient une référence sur elle-même afin de définir le nœud parent. Cela me permettra par la suite d'utiliser une fonction récursive pour l'affichage.

6.1.7 Table Files

Cette table permet aux utilisateurs de lier un fichier à une tâche. Les fichiers sont stockés sur l'ordinateur et non dans la base. L'attribut *filename* permet de retrouver le fichier grâce à son chemin d'accès. Un attribut *description* permet de définir le rôle de ce fichier dans la tâche.

6.1.8 Table FileTypes

Cette table sert de constante.

Elle permet de garder en mémoire les types de fichiers.

6.1.9 Table Activities

Cette table permet d'avoir des activités liées aux tâches dans le projet. Elle permet de garder une trace sur les actions lors du travail sur un projet.

6.1.10 Table Type

Cette table sert de constante.

Elle permet de garder en mémoire si la tâche créée est une fonctionnalité ou si cette tâche permet la résolution d'un problème. Elle ne contient que très peu d'enregistrements.

6.1.11 Table Comments

Cette table permet aux utilisateurs de pouvoir commenter une tâche. Les commentaires ont une date et une heure afin de pouvoir les ordonner et un utilisateur y sera assigné pour savoir à qui s'adresser.

6.1.12 Table Priorities

Cette table sert de constante.

Elle permet de garder en mémoire les ordres de priorité et ne contient que très peu d'enregistrements.

6.1.13 Table Checklists

Cette table permet de créer une checklist qui servira à fractionner les tâches d'une UserStory. Un nom est attribué par les utilisateurs pour donner un nom à la liste.

6.1.14 Table ChecklistItems

Cette table permet de créer un objet dans la checklist liée. L'attribut *done* permet de savoir si la tâche créée par cet enregistrement est effectuée. L'attribut *name* permet d'afficher dans les vues un identifiant déterminé par l'utilisateur.

Une table de liaison la reliera avec la table *users* afin de pouvoir attribuer des utilisateurs aux différentes tâches que cette table va créer.

6.2 Modèle de classes

Afin de séparer le projet au mieux, j'ai décidé d'utiliser une approche Modèle-Vue-Contrôleur. Nous avons d'abord les classes de données, représentées dans la Figure 27. Ensuite, les contrôleurs représentés dans la Figure 28. Finalement, les classes de la vue sont représentés dans la Figure 29.

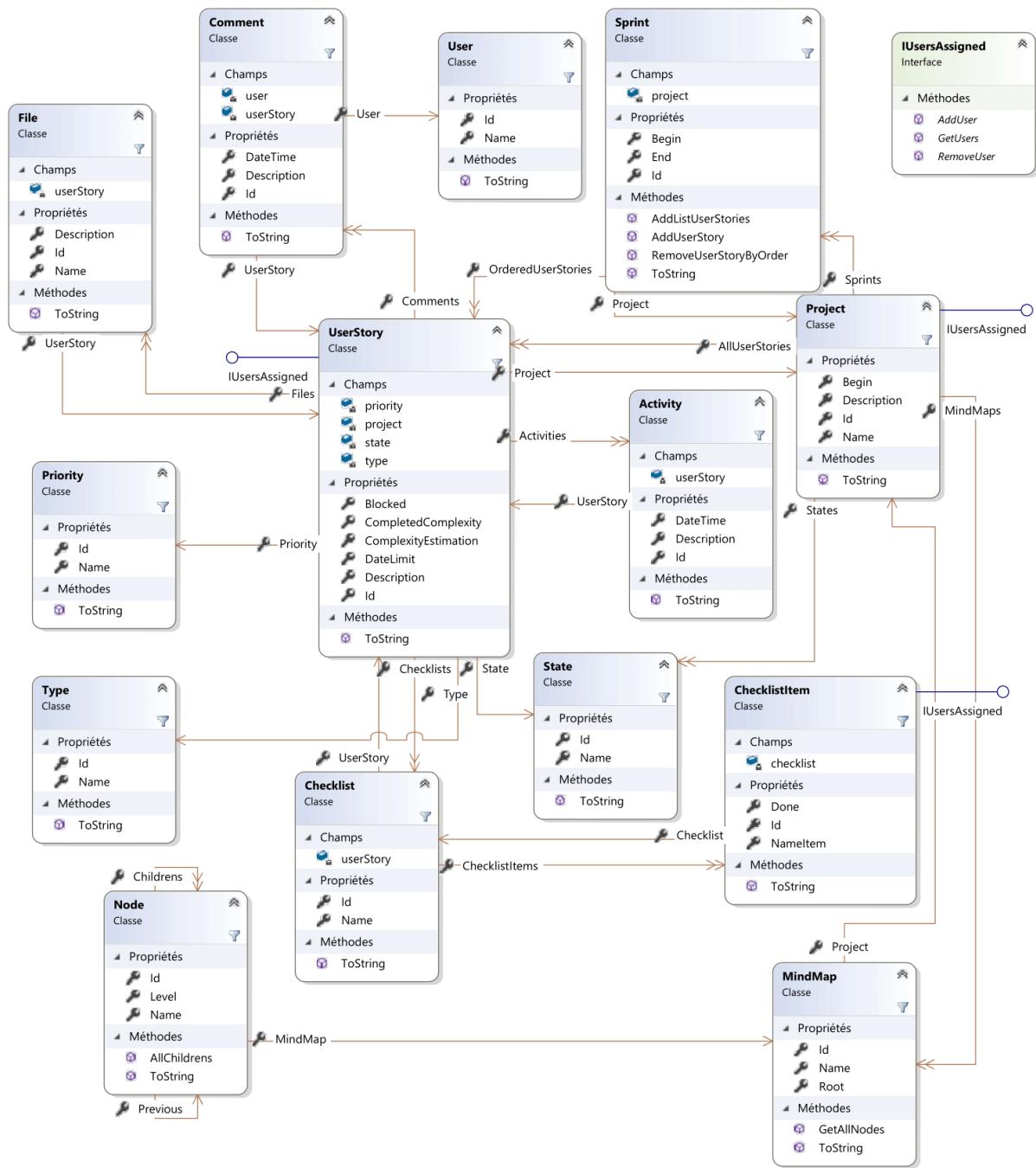


Figure 27 Diagramme des classes de données

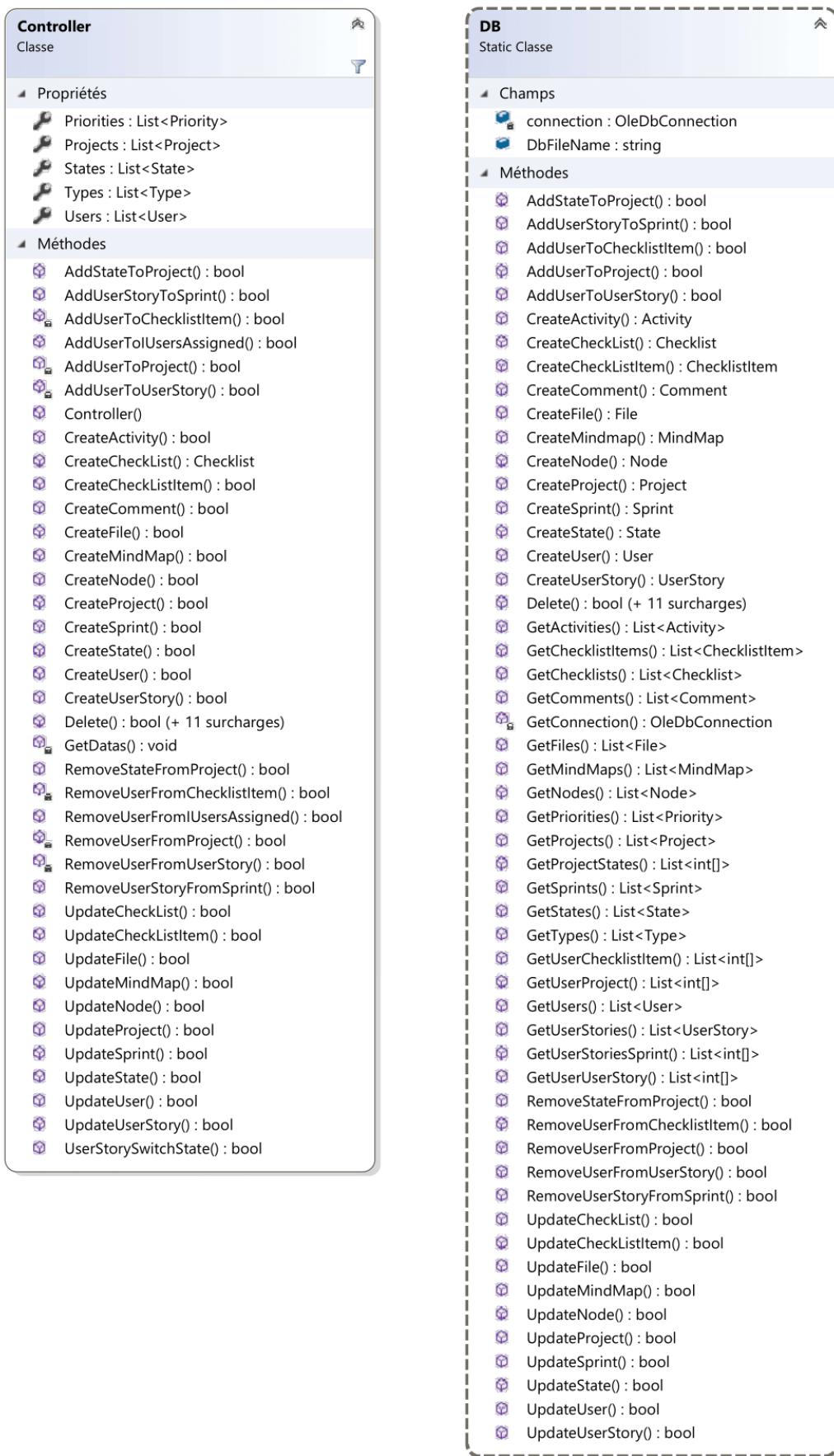


Figure 28 Diagramme des classes de contrôle

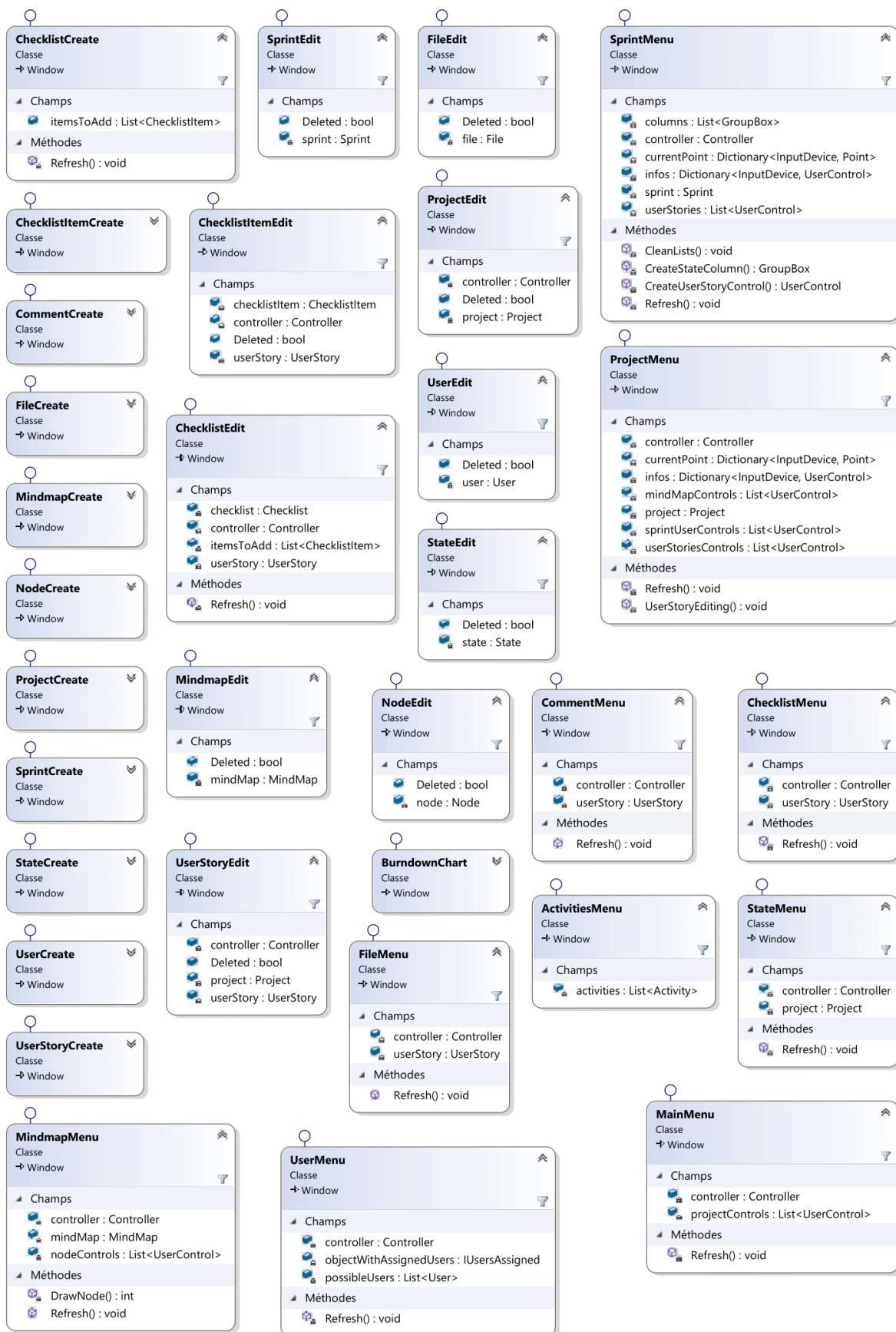


Figure 29 Diagramme des classes de vues

6.3 Apports personnels

Un projet complet de ce niveau ne peut pas être effectué dans son entiereté par une seule personne dans le temps qu'il m'a été donné. C'est pourquoi j'ai utilisé des outils préfabriqués afin de réaliser certaines tâches tel que draw.io pour réaliser certains diagrammes et Balsamiq pour réaliser les maquettes d'écrans. J'ai également utilisé les outils fournis avec Visual Studio pour les diagrammes de classes.

Pour ce qui est du code, je me suis basé sur WPF et sa gestion des évènements afin d'utiliser mes propres méthodes pour les évènements.

J'ai donc réalisé dans ce projet tout ce qui se situe dans les classes présentées dans le chapitre 6.2 Modèle de classes.

Pour ce qui est de la base de données, je l'ai réalisé de la conception à l'implémentation dans le programme en passant par la création dans Access.

6.4 Communications avec la base de données

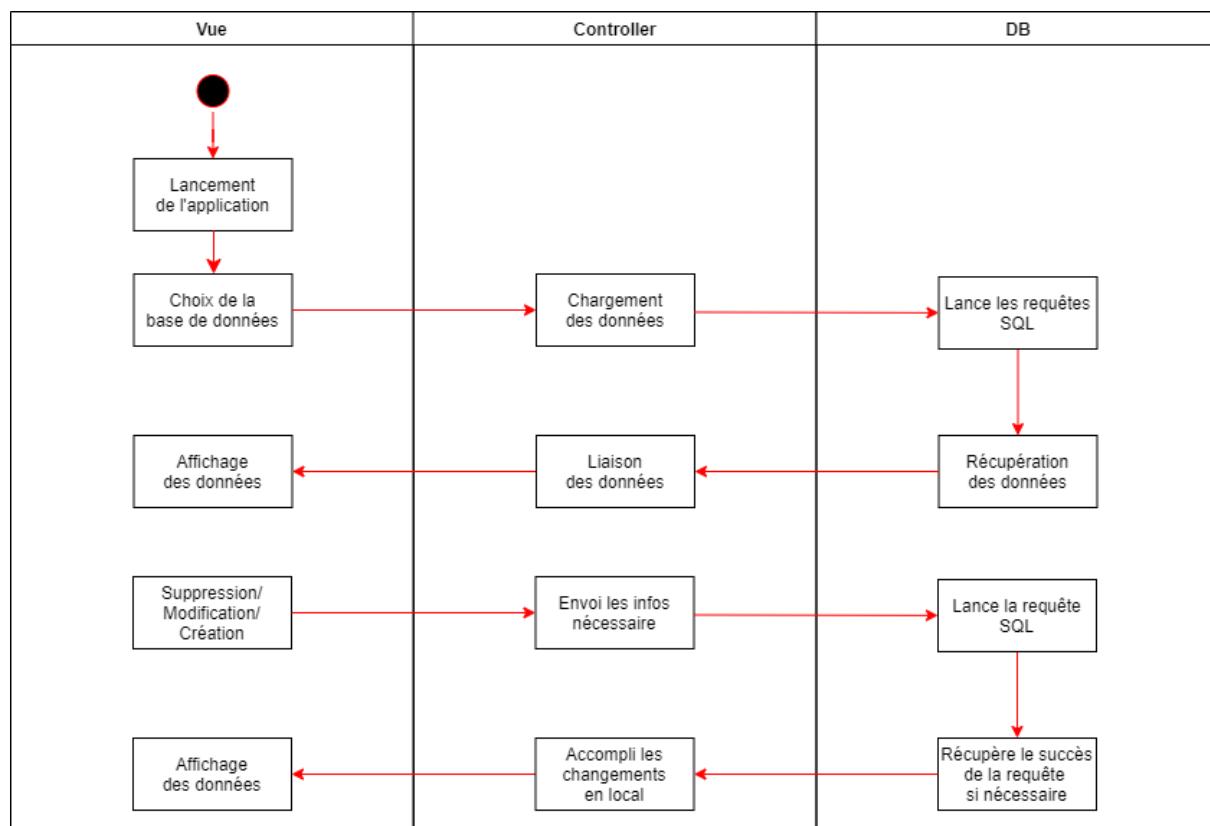


Figure 30 Diagramme de séquence de communication à la base de données

Afin de ne pas ralentir le projet avec les requêtes SQL d'acquisition de données, j'ai décidé de récupérer toutes les données au lancement de l'application. Cela permet d'avoir déjà toutes les données et de pouvoir traverser les vues d'une façon beaucoup plus fluide.

Cependant, afin de ne pas avoir de temps morts durant lesquels des milliers d'enregistrements se créent et se modifient, je n'ai pas trouvé pertinent de couper l'accès à la base de données totalement. J'accède à la base de données lors de la création, suppression et modification des enregistrements afin d'éviter des soucis en cas d'erreur fatale ou de coupure de courant.

6.5 Liaison Access

```
using System.Data.OleDb;

OleDbConnection connection = new
OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" + DbFileName
+ ";Persist Security Info=False;");

//Initialize variables
OleDbCommand cmd;
bool result;
//Open database, build sql statement and prepare
connection.Open();
cmd = connection.CreateCommand();
cmd.CommandText = "UPDATE TComments SET deletedFlag = ? WHERE IdComment = ?;";
cmd.Parameters.Add("deletedFlag", OleDbType.Boolean);
cmd.Parameters.Add("IdComment", OleDbType.Integer);
cmd.Parameters[0].Value = true;
cmd.Parameters[1].Value = comment.Id;
//Execute sql statement
cmd.Prepare();
result = cmd.ExecuteNonQuery() == 1;

//Close database
connection.Close();
return result;
```

Figure 31 Code d'utilisation de la liaison avec Access

La liaison avec une base de données Access utilise la classe de données de C#, OleDb. Il faut pour cela déclarer l'utilisation de cette classe dans les déclarations d'usage

Il faut ensuite créer un objet OleDbConnection qui permet d'établir la liaison entre l'application C# et la base de données Access.

J'ai inséré pour l'exemple la suppression d'un commentaire dans la base de données avec la suppression douce. Je commence par ouvrir la connexion. Je définis ensuite la requête SQL dans une chaîne de caractères.

J'ai ensuite créé les paramètres ainsi que leur type. Je leur attribue des valeurs et, finalement, j'exécute la requête et récupère le résultat (Dans ce cas, le nombre de lignes affecté égal à 1). A la fin je ferme la connexion et je retourne le résultat.

6.6 Glisser-déposer multipoint

Afin de réaliser le glisser-déposer, j'ai besoin de tableau associatifs qui se trouvent en C# sous la forme des dictionnaires. J'utilise une liste avec une position comme valeur et une avec les informations. (Voir Figure 32)

```
private readonly Dictionary<InputDevice, Point> currentPoint = new Dictionary<InputDevice, Point>();
private readonly Dictionary<InputDevice, UserControl> infos = new Dictionary<InputDevice, UserControl>();
```

Figure 32 Déclaration des variables nécessaire au glisser-déposer

Lors de l'appui sur un élément pouvant être glissé, je vérifie si l'identifiant est déjà existant car le programme peut réutiliser le même identifiant après avoir disparu. Après m'être assuré que l'identifiant n'existe pas déjà, je rajoute un enregistrement dans les listes. (Voir Figure 33)

```
private void UserStory_PreviewTouchDown(object sender, TouchEventArgs e)
{
    if (currentPoint.ContainsKey(e.Device))
    {
        currentPoint.Remove(e.Device);
        infos.Remove(e.Device);
    }

    currentPoint.Add(e.Device, e.GetTouchPoint(null).Position);
    infos.Add(e.Device, sender as UserControl);
}
```

Figure 33 Création d'un nouvel enregistrement dans le glisser-déposer

Lors du déplacement de l'élément, je mets à jour la position de l'élément. Cet exemple venant de la fenêtre de sprint, je mets à jour la taille de toutes les bordures à la taille par défaut. Ensuite, je parcours tous les éléments existants pour augmenter la taille des colonnes qu'un élément survole. Cela permet de voir visuellement la destination. (Voir Figure 34)

```
private void SprintMenu_PreviewTouchMove(object sender, TouchEventArgs e)
{
    //Update position of point
    if (currentPoint.ContainsKey(e.Device))
    {
        currentPoint[e.Device] = e.GetTouchPoint(null).Position;
    }
    //Set border thickness to default
    foreach (GroupBox col in columns)
    {
        col.BorderThickness = new Thickness(1);
    }
    //Verify each points to change border thickness if in bounds
    foreach (KeyValuePair<InputDevice, Point> keyValuePair in currentPoint)
    {
        Point p = keyValuePair.Value;
        foreach (GroupBox col in columns)
        {
            double leftBound = Canvas.GetLeft(col);
            double rightBound = leftBound + col.Width;
            double topBound = Canvas.GetTop(col);
            double bottomBound = topBound + col.Height;

            if (p.X > leftBound && p.X < rightBound &&
                p.Y > topBound && p.Y < bottomBound)
            {
                col.BorderThickness = new Thickness(3);
            }
        }
    }
}
```

Figure 34 Déplacement d'un élément

Finalement, lors du relâchement, je vérifie tout d'abord que l'identifiant de l'élément existe dans la liste. Ensuite, je récupère la colonne contenant la position de l'élément. Si une colonne est détectée, je change l'user story de colonne. Dans tous les cas, je finis en supprimant l'élément des listes et en rechargeant la fenêtre. (Voir Figure 35)

```

private void SprintMenu_PreviewTouchUp(object sender, TouchEventArgs e)
{
    if (currentPoint.ContainsKey(e.Device))
    {
        //Search the contained groupbox
        Point p = e.GetTouchPoint(null).Position;
        GroupBox gbxState = null;
        foreach (GroupBox col in columns)
        {
            double leftBound = Canvas.GetLeft(col);
            double rightBound = leftBound + col.Width;
            double topBound = Canvas.GetTop(col);
            double bottomBound = topBound + col.Height;
            if (p.X > leftBound && p.X < rightBound &&
                p.Y > topBound && p.Y < bottomBound)
            {
                gbxState = col;
                break;
            }
        }
        //if release in a groupbox, change state
        if (gbxState != null)
        {
            gbxState.BorderThickness = new Thickness(1);

            State state = gbxState.Tag as State;
            UserStory userStory = (infos[e.Device] as UserControl).Tag as UserStory;
            controller.UserStorySwitchState(userStory, state);
        }
        currentPoint.Remove(e.Device);
        infos.Remove(e.Device);
        Refresh();
    }
}

```

Figure 35 Relâchement de l'appui

7 Plan de tests

7.1 Test d'interface

N°	Description
Démarrage	
1	L'application compile et démarre.
2	L'application récupère les données de la base de données fournie.
3	L'application renvoie une erreur si la base de données n'est pas correcte.

N°	Description
Menu principal	
4	Les projets s'affichent sans erreur
5	Cliquer sur un projet amène sur la fenêtre de projet
6	Cliquer sur le bouton « + » amène sur le pop-up de création de projet
7	Cliquer sur le bouton « < » ferme l'application
7.5	Le pop-up crée en cas de confirmation et ne fait rien en cas d'annulation

N°	Description
Fenêtre de projet	
8	Cliquer sur une User Story ouvre la fenêtre de modification d'user stories
9	Cliquer sur un sprint ouvre la fenêtre de sprint
10	Cliquer sur le bouton « < » ramène au menu principal
11	Cliquer sur le bouton « Modifier » ouvre le pop-up de modification de projet
12	Cliquer sur le bouton « + » des sprint ouvre le pop-up de création de sprint
12.5	Le pop-up crée en cas de confirmation et ne fait rien en cas d'annulation
13	Cliquer sur le bouton « + » des user stories ouvre le pop-up de création de sprint
13.5	Le pop-up crée en cas de confirmation et ne fait rien en cas d'annulation

N°	Description
Fenêtre de sprint	
14	Le glisser déposer d'une User Story la change d'état dans la base et visuellement
15	Cliquer sur le bouton « Calendrier » ouvre la fenêtre du burndown chart
16	Cliquer sur le bouton « < » ramène à la fenêtre de projet
17	Le menu contextuel permet d'ajouter une user story
18	Cliquer sur une User story ouvre la fenêtre de modification d'user story

N°	Description
Fenêtre de burndown chart	
19	Affiche en rouge la droite idéale de progression de sprint
20	Affiche en noir la progression du sprint actuelle

N°	Description
Fenêtre de modification d'user story	
21	Cliquer sur le bouton « Fichiers » ouvre la fenêtre des fichiers
22	Cliquer sur le bouton « Commentaires » ouvre la fenêtre des commentaires
23	Cliquer sur le bouton « Checklists » ouvre la fenêtre des checklists
24	Cliquer sur le bouton « Activités » ouvre la fenêtre des activités
25	Cliquer sur le bouton « Utilisateurs assignés » ouvre la fenêtre des utilisateurs

N°	Description
Fenêtre de modification de projet	
26	Cliquer sur le bouton « Utilisateurs assignés » ouvre la fenêtre des utilisateurs
27	Cliquer sur le bouton « Colonnes » ouvre la fenêtre des états

N°	Description
Fenêtre des Fichiers	
28	Cliquer sur un Fichier ouvre le pop-up de modification de fichier
28.5	Le pop-up modifie en cas de confirmation et ne fait rien en cas d'annulation
29	Cliquer sur le bouton « + » ouvre le pop-up d'association de fichier
29.5	Le pop-up crée en cas de confirmation et ne fait rien en cas d'annulation
30	Cliquer sur le bouton « Retour » ferme la fenêtre

N°	Description
Fenêtre des commentaires	
31	Cliquer sur le bouton « + » ouvre le pop-up de création de commentaire
31.5	Le pop-up crée en cas de confirmation et ne fait rien en cas d'annulation
32	Cliquer sur le bouton « Retour » ferme la fenêtre

N°	Description
Fenêtre des checklists	
33	Cliquer sur une liste ouvre le pop-up de modification de liste
33.5	Le pop-up modifie en cas de confirmation et ne fait rien en cas d'annulation
34	Cliquer sur le bouton « + » ouvre le pop-up de création de liste.
34.5	Le pop-up crée en cas de confirmation et ne fait rien en cas d'annulation
35	Cliquer sur le bouton « Retour » ferme la fenêtre

N°	Description
Fenêtre des activités	
36	Cliquer sur le bouton « Retour » ferme la fenêtre

N°	Description
Fenêtre des états	
37	Cliquer sur le bouton « + » ouvre le pop-up de création d'état
37.5	Le pop-up crée en cas de confirmation et ne fait rien en cas d'annulation
38	Cliquer sur le bouton « > » fait passer l'élément sélectionné de gauche à droite
39	Cliquer sur le bouton « < » fait passer l'élément sélectionné de droite à gauche
40	Cliquer sur le bouton « Enregistrer » ferme la fenêtre
41	La fermeture de la fenêtre enregistre les modifications

N°	Description
Fenêtre des utilisateurs	
42	Cliquer sur le bouton « + » ouvre le pop-up de création d'état
42.5	Le pop-up crée en cas de confirmation et ne fait rien en cas d'annulation
43	Cliquer sur le bouton « > » fait passer l'élément sélectionné de gauche à droite
44	Cliquer sur le bouton « < » fait passer l'élément sélectionné de droite à gauche
45	Cliquer sur le bouton « Enregistrer » ferme la fenêtre
46	La fermeture de la fenêtre enregistre les modifications

7.2 Tests unitaires

J'ai créé des tests seulement pour les classes de contrôle. J'ai choisi de ne pas implémenter de fichier de test pour chaque classe car la classe de test « ControllerTests » vérifie les intégrités de chaque classe en même temps que leurs tests.

►	ControllerTests (16)	1.2 min
✓	CRDActivity	5 s
✓	CRDComment	5 s
✓	CRDProjectStatesTest	4.3 s
✓	CRDUserAssignedTest	6.6 s
✓	CRDUserStoriesSprintTest	5.2 s
✓	CRUDChecklist	4.7 s
✓	CRUDChecklistItem	5.1 s
✓	CRUDFile	4.8 s
✓	CRUDMindMap	4.1 s
✓	CRUDNode	4.5 s
✓	CRUDProject	3.7 s
✓	CRUDSprint	4.3 s
✓	CRUDState	3.8 s
✓	CRUDUser	4 s
✓	CRUDUserStoryTest	4.4 s
✓	ControllerTest	5.3 s

Figure 36 Résultat des tests sur la classe Controller

Comme vous pouvez le voir sur la Figure 36, Les tests ont été ordonnés par classe. J'ai préféré les ordonnées comme cela afin de ne pas avoir beaucoup de tests. Comme vous l'aurez peut-être remarqué, certaines classes ont besoin d'un CRUD complet alors que d'autres n'utilisent qu'un CRD.

Finalement, La méthode « ControllerTest » vérifie que le contrôleur récupère les données.

DBTests (19)	17.9 s
CRDActivity	996 ms
CRDComment	752 ms
CRDProjectStatesTest	1.1 s
CRDUserChecklistItemTest	2.2 s
CRDUserProjectTest	1.1 s
CRDUserStoriesSprintTest	1.8 s
CRDUserUserStoryTest	1.7 s
CRUDChecklist	760 ms
CRUDChecklistItem	749 ms
CRUDFile	755 ms
CRUDMindMap	744 ms
CRUDNode	1.3 s
CRUDProject	1 s
CRUDSprint	744 ms
CRUDState	305 ms
CRUDUser	301 ms
CRUDUserStoryTest	1.2 s
GetPrioritiesTest	147 ms
GetTypesTest	151 ms

Figure 37 Résultat des tests sur la classe DB

Comme vous pouvez le voir sur la Figure 37, j'ai utilisé des tests couvrant les opérations susceptibles de survenir réellement. Il y'a par exemple les activités qui sont CRD. Ceci montre que le test comprend la création, la lecture et la suppression sur cette table mais pas de modifications. En effet, la table activité ne sera jamais utilisée pour de la modification car cette option n'est pas pertinente.

Finalement, les méthodes dont le nom commence par « Get » sont les tables de constantes pour les types des User Stories ou pour les priorités qu'il est possible de donner.

8 Conclusion

J'ai rencontré des difficultés inédites durant ce projet. En effet, le confinement dû au Covid-19 m'a empêché de pouvoir travaillé sur l'outil voulu. J'ai eu la chance que M. Garcia m'apporte un écran tactile afin de palier ce problème. De plus, le nombre de distractions existantes au sein de mon lieu de vie est élevé. J'ai donc dû me discipliner afin de travailler au mieux sur le projet. Je pense avoir réussi à fournir une application de qualité malgré ce point.

De plus, l'utilisation de WPF m'était totalement inconnue. J'ai bien sûr essayé de créer une application avant. Ce qui m'a aidé à surmonter cette difficulté fut les différents sites tel que microsoft ou stackoverflow. Beaucoup de gens ont déjà eu des problèmes similaires à ceux que j'ai eu et la communauté répondait avec des explications claires qui m'ont permis de mener à bien ce projet.

Ce projet n'est pas parfait et les rajouts possibles sont nombreux. On peut par exemple penser à l'ajout d'une gestion du projet dans le temps via un affichage sous la forme de gantt. Un changement afin d'ajouter le glisser-déposer à la souris pourrait également être réalisé. Il y a encore sûrement beaucoup de fonctionnalités auxquels je n'ai même pas pensé.

Finalement, je suis globalement content de mon projet. J'ai réussi à produire une application fonctionnelle malgré les difficultés rencontrées. J'ai même pu rajouter une fonctionnalité qui n'était pas prévue de base, la création de mindmap.

9 Table des figures

Figure 1 Capture d'écran de trello	5
Figure 2 Capture d'écran de JIRA.....	6
Figure 3 Capture d'écran de Ubikeey.....	6
Figure 4 Capture d'écran de Kantree	7
Figure 5 Planning initial.....	8
Figure 6 Planning final.....	9
Figure 7 Maquette de la fenêtre de sélection de projet.....	11
Figure 8 Maquette de la fenêtre de projet	12
Figure 9 Maquette de la fenêtre de sprint	13
Figure 10 Maquette de la fenêtre du Burndown Chart	14
Figure 11 Maquette de la fenêtre des utilisateurs	14
Figure 12 Maquette de la fenêtre des états.....	15
Figure 13 Maquette de la fenêtre des activités.....	16
Figure 14 Maquette de la fenêtre des listes	16
Figure 15 Maquette de la fenêtre des commentaires	17
Figure 16 Maquette de la fenêtre de fichiers.....	18
Figure 17 Maquette du popup de création de projet	18
Figure 18 Maquette du popup de création de UserStory	19
Figure 19 Maquette du popup de création de sprint	19
Figure 20 Maquette de pop-up de création de commentaire	20
Figure 21 Maquette de pop-up de création d'utilisateur	20
Figure 22 Maquette de pop-up de création d'état.....	21
Figure 23 Maquette de pop-up de création checklist.....	21
Figure 24 Maquette de pop-up de création fichier	22
Figure 25 Maquette écran du mindmap	23
Figure 26 Modèle Conceptuel de Données	24
Figure 27 Diagramme des classes de données	27
Figure 28 Diagramme des classes de contrôle	28
Figure 29 Diagramme des classes de vues	29
Figure 30 Diagramme de séquence de communication à la base de données	30
Figure 31 Code d'utilisation de la liaison avec Access	31
Figure 32 Déclaration des variables nécessaire au glisser-déposer	31
Figure 33 Création d'un nouvel enregistrement dans le glisser-déposer	32
Figure 34 Déplacement d'un élément	33
Figure 35 Relâchement de l'appui	34
Figure 36 Résultat des tests sur la classe Controller.....	37
Figure 37 Résultat des tests sur la classe DB	38

10 Annexes

Scrum-o-wall Journal de bord

06.04

Tâches à faire

- Créer le versionning
- Commencer la structure de la documentation
- Redéfinir le cahier des charges

Liens utiles et idées

Recherche de ressources pour faire du multi touch <https://docs.microsoft.com/fr-fr/dotnet/framework/wpf/advanced/walkthrough-creating-your-first-touch-application?redirectedfrom=MSDN> : Permet de changer la taille, tourner et déplacer en utilisant plusieurs doigts Il faut mettre tout ça dans une classe pour faire des post-it avec et pas que ce soit la fenêtre qui s'en occupe

<https://thedigitalprojectmanager.com/fr/meilleurs-outils-scrum/> : Pour l'étude d'opportunité

Tâches accomplies

- Versioning créé
- Première structure de la documentation effectuée
- Nécessite encore une restructuration de la base de données

07.04

Tâches à faire

- Restructuration de la base de données
- Création de la base de données
- Documentation de la base de données

Liens utiles et idées

Je pourrais intégrer un Mindmap en créant une table à chaque mindmap et en utilisant une table pour lier les tables avec le projet. Cependant, l'optimisation risque d'être horrible. C'est soit ça, soit avoir une méga table qui contiendrait tous les mindmap

Tâches accomplies

- Restructuration accomplies
- Base de donnée créée dans un fichier accdb
- Documentation mise à jour pour la base de données

08.04

Tâches à faire

- Création des maquettes
- Documentation des maquettes

Liens utiles et idées

J'ai fais les maquettes et la documentation de toutes les maquettes. J'ai même fais les maquettes si jamais je réussissais à faire le mindmap et le gantt (qui s'avèrera peut-être une visualisation des tâches accomplies durant le sprint plutôt qu'un vrai gantt)

Tâches accomplies

- Maquettes créé
- Maquettes documentées sauf gantt et mindmap

09.04 (Vacances)

Pendant les vacances, je vais continuer à travailler mais à un rythme moins élevé. J'ai pas mal de trucs à faire chez moi donc je vais en profiter pour effectuer ces tâches ménagères.

Tâches à faire

- Début de la création du poster
- Création d'un icône

Liens utiles et idées

J'ai du raccourcir l'écriture du projet afin qu'il rentre dans un icone carré sans pour autant avoir de préjudices sur la lisibilité.

Tâches accomplies

- Icône créé pour github et pour l'application
- Poster commencé

14.04

Tâches à faire

- Finir le poster
- Créer le résumé

Liens utiles et idées

J'ai décidé de mettre des post-it sur le poster afin de représenter la méthode agile ainsi qu'un mindmap comme image d'arrière plan afin de faire penser à la reflexion qu'on doit avoir en permanence lors du développement agile. J'ai voulu garder le poster assez sobre et ne pas rajouter d'informations pour ne pas surcharger l'image. Je ne sais pas si le résumé est correct. Je devrais demander conseil à Mme. Terrier.

Tâches accomplies

- Poster
- Premier résumé fini

15.04

Tâches à faire

- Recherche sur les existants

Liens utiles et idées

- monday.com : https://monday.com/lp/mb/dpm/scrum/?utm_source=mb&utm_campaign=dpm&utm_medium=scrum%20
- JIRA : <https://www.atlassian.com/software/jira?r=sct>

Spécifique aux écrans interactifs <https://www.speechi.net/fr/2019/06/04/comparatif-des-outils-de-visual-management-pour-ecran-interactif/> - Ubikey : <https://www.ubikey.fr/>

- Trello : <https://trello.com/fr>
- Kantee : <https://kantee.io/fr/>
- Wrike : <https://www.wrike.com/fr/>

Tâches accomplies

- Trouver plein d'applications pareils

20.04

Tâches à faire

- Analyser les existants
- Créer les classes

Liens utiles et idées

J'ai réduit pas mal les existant pour garder que les plus pertinents. J'ai déjà créer les fonctions pour altérer les listes des classes.

Tâches accomplies

- Existant analyser
- Classes créées

21.04

Tâches à faire

-Faire une vue

Liens utiles et idées

En fait il faudrait que je fasse le controller avant de faire les vues. Je vais donc commencer le controller J'ai eu une erreur "Le fournisseur 'Microsoft.ACE.OLEDB.12.0' n'est pas inscrit sur l'ordinateur local". Je pense que je ne dois pas avoir les drivers. Après recherches, j'ai trouvé un lien en ligne : <https://www.microsoft.com/fr-FR/download/details.aspx?id=39358> De plus, Visual Studio à choisi de lancer l'application de préférence en 32 bits alors que l'engine de base de données est en 64 bits. J'ai donc du désactivé la préférence aux 32 bits dans les options de Build de l'application

Tâches accomplies

- Début de la vue (Manque la création dynamique des projets et de leurs liens respectifs)
- Singlotron de la Database fonctionnelle

22.04

Tâches à faire

- Finir le contrôleur pour la vue des projets
- Finir la vue des projets

Liens utiles et idées

Je galère à trouver un algorithme pour centrer verticalement les frames des projets. De plus, je risque de rencontrer un problème si il y a vraiment trop de projets dans la database. Je suis donc parti du haut de la fenêtre et j'ai mis le tout dans un ScrollViewer pour qu'une scrollbar s'affiche automatiquement quand il y a trop de contenu. Il faut encore que je crée le popup de création et j'aurais fini pour la vue des projets.

Tâches accomplies

- Fais la création et la récupération des projets
- Première vue créée avec le popup de création

23.04

Tâches à faire

- Créer la forme principale des autres vues
- Créer une méthode controller pour avoir toutes les infos des projets
- Mise à jour du planning actuel

Liens utiles et idées

Il n'existe pas de numeric updown par défaut dans WPF. Mais il existe un package nuget qui l'ajoute (Source : <https://stackoverflow.com/questions/841293/where-is-the-wpf-numeric-updown-control>) Je sais pas si je vais utiliser cela car il ne permet que peu de choses de par sa licence. Je vais chercher une autre solution. Finalement je vais juste filtrer les entrées utilisateurs et mettre un textbox. C'est le plus simple à faire sans plugin. J'aimerais quand même avoir un petit clavier virtuel qui respecterait le multi-point.

Tâches accomplies

- Planning mis à jour
- Le contrôleur récupère toutes les infos au lancement de l'application
- Les vues du sprint et du backlog ne sont pas finies (sans compter le gantt et le mindmap)

24.04

Tâches à faire

- Finir les vues
- Connecter les vues entre elles

Liens utiles et idées

J'ai pensé à peut-être une protection de projet par mot de passe.

Tâches accomplies

- Vues finies
- La connection avec la vue du sprint a un problème. Il ne trouve pas le groupbox

27.04

Tâches à faire

- Régler le problème de la vue du sprint
- Faire les méthodes d'ajout

Liens utiles et idées

Le problème était que je cherchais le nom sans avoir enregistré le nom : <https://docs.microsoft.com/en-us/dotnet/api/system.windows.frameworkelement.registername?view=netcore-3.1> Je viens de penser au rajout de colonne et je remarque qu'il me manque des vues. Il me faut un popup de création de colonnes et un popup de gestion de colonnes

Tâches accomplies

- Problème de la vue du sprint réglé
- Export des méthodes d'accès à la DB dans la classe DB
- Modification de l'orthographe de la doc
- Analyse et priorisation des fonctions du trello

28.04

Tâches à faire

- Approfondir l'analyse approfondie de Trello
- Restructurer la documentation
- Modifier la base de données et la documentation pour s'adapter à l'analyse de Trello.

Liens utiles et idées

Tâches accomplies

- Analyse sur Trello approfondie
- Documentation restructurée
- Changement du Modèle Conceptuel de Données et documentation des changements

29.04

Tâches à faire

- Modifier la base de données
- Créer/Modifier les classes

Liens utiles et idées

Je ne trouvais pas le diagramme de classe mais après une recherche il fallait l'installer par le Visual Studio Installer

Tâches accomplies

- Base de données modifiée
- Création des classes et modification des existantes
- Création du diagramme de classe

30.04

Tâches à faire

- Modification des maquettes existantes
- Refaire des maquettes pour les fenêtres manquantes
- Faire un plan de tests

Liens utiles et idées

Tâches accomplies

- Modification des maquettes
- Maquettes pour les fenêtres manquantes
- Début du plan de test

01.05

Tâches à faire

- Finir le plan de test

- Faire la réunion avec Mme. Terrier
- Faire le visuel des maquettes

Liens utiles et idées

Pour la gestion des listes il faudra que je reproduise ça en code :

```
<Border BorderBrush="Black" BorderThickness="1" Width="408" >
    <Grid x:Name="list1" >
        <Grid.ColumnDefinitions>
            <ColumnDefinition Width="30*" />
            <ColumnDefinition Width="67*" />
        </Grid.ColumnDefinitions>
        <TextBlock VerticalAlignment="Top" TextWrapping="Wrap" Margin="10,10,10,0" TextAlignment="Right"></TextBlock>
        <ListView Grid.Column="1" VerticalAlignment="Top" x:Name="listItems" Margin="10,10,10,10" Height="76" >
            <CheckBox IsChecked="True">asdasd</CheckBox>
            <CheckBox IsChecked="False">asdasd</CheckBox>
            <CheckBox >asdasd</CheckBox>
        </ListView>
    </Grid>
</Border>
```

Tâches accomplies

- Plan de test des vues
- Réunion avec Mme.Terrier
- Fais la moitié des visuels

04.05

Tâches à faire

- Finir les visuels
- Créer les constructeurs des vues
- Placer les valeurs dans le visuel

Liens utiles et idées

Trouvé de quoi faire la ligne pour le burndown chart <https://www.c-sharpcorner.com/UploadFile/mahesh/polyline-in-wpf/>

Tâches accomplies

- Finir les visuels
- Fais les vues suivantes ; Burndown Chart, ChecklistCreate, ActivitesMenus, CheckListItemCreate, ChecklistMenu, CommentCreate, CommentsMenu, MindMapCreate, CommentCreate, CommentMenu

05.05

Tâches à faire

- Faire une partie des vues restantes

Liens utiles et idées

Tâches accomplies

- Fini les vues : FileCreate, FileEdit, FileMenu, ProjectEdit, StateCreate, StateMenu, UserCreate, UserEdit, UserStoryCreate
- Il reste UserStoryEdit et UserMenu

06.05

Tâches à faire

- Faire les vues restantes
- Corriger le poster

Liens utiles et idées

Afin de pouvoir utiliser le menu des utilisateurs j'ai du créer une interface car plusieurs classes différentes l'utiliseront. J'ai découvert grâce à cela que la version 8.0 de C# permet d'implémenter les propriétés. Cependant, vu que j'utilise la version 7.3, j'ai dû créer des méthodes pour altérer la liste d'utilisateurs.

Tâches accomplies

- Fini les vues : UserStoryEdit et UserMenu
- Création d'une interface
- Rajout du langage de programmation sur le poster

07.05

Tâches à faire

- Refaire la documentation sur les maquettes écrans
- Refaire la documentation des classes

Liens utiles et idées

Important de citer que j'ai voulu utiliser du déconnecté Expliquer la stratégie Expliquer la notice de connection à Access Pourquoi j'ai fait une classe d'interface, pourquoi j'ai fait ci et ça. Les challenges du WPF. Ce qui est lié à l'écran tactile.

Tâches accomplies

- Documentation sur les maquettes écrans refaite
- Les diagrammes de classes ont été ajoutés et nécessitent une documentation plus poussée
- Correction de bugs empêchant la compilation

08.05

Tâches à faire

- Implémenter les méthodes de la base de données

Liens utiles et idées

J'ai eu un problème sur la base de données. Il y avait une erreur lors de l'ajout d'un champ pouvant être nul. Après une recherche, j'ai découvert grâce à cette source : <https://www.c-sharpcorner.com/article/enter-null-values-for-datetime-column-of-sql-server/> qu'une classe DBNull possèdait une constante pour les valeurs nulles des bases de données.

Tâches accomplies

- Implémentation des méthodes de la base de données

11.05

Tâches à faire

- Rajouter la gestion de l'écran tactile.
- Inclure le drag'n'drop sur ProjectMenu

Liens utiles et idées

J'ai remarqué qu'il me manquait une vue la dernière fois. Il me manque ChecklistItemEdit.

En plus, il faut que je commence à m'intéresser au drag'n'drop en wpf. J'ai trouvé ce post sur stackoverflow qui pourrait m'en apprendre plus : <https://stackoverflow.com/questions/11306194/wpf-dragenter-between-two-canvases-not-firing> Je vais essayer de faire un drag and drop sur le menu du projet.

Le post stackoverflow renvoie sur un tutoriel de wpftutorial : <http://wpftutorial.net/DragAndDrop.html>.

J'ai un problème pour la liaison du sprint avec les user stories "INSERT INTO TUserStoriesSprint (IdUserStory,IdSprint,Order) VALUES (2,3,1);," ne fonctionne pas avec erreur de syntaxe mais "INSERT INTO TUserStoriesSprint VALUES (2,3,1);" fonctionne.

Après avoir cherché beaucoup trop de temps pour que ce soit raisonnable, j'ai trouvé le problème. Le mot order est réservé par SQL pour le "ORDER BY". j'ai donc renommé le nom du champ par "OrderUserStory".

Tâches accomplies

- Gestion de l'écran tactile accomplies
- Le drag'n'drop a été ajouté sur ProjectMenu

12.05

Tâches à faire

- Inclure le drag'n'drop sur SprintMenu
- Rendre le drag'n'drop plus évident

Liens utiles et idées

J'ai effectué le drag'n'drop mais les user controller ne sont pas effacés. Il faudrait que je mette à jour uniquement le bon UserControl ou que je remette à jour correctement tous les éléments de la fenêtre.

J'ai choisi la deuxième option et j'ai du refaire totalement la mise à jour de la vue. Cela m'a pris plus de temps que prévu et je n'ai pas le temps de faire ma deuxième activité.

Tâches accomplies

- Drag'n'drop sur sprint menu effectué

13.05

Tâches à faire

- Faire le résumé
- Faire l'abstract
- Rendre le résumé et l'abstract

Liens utiles et idées

Ayant un vocabulaire en français plus fourni que mon vocabulaire anglais, j'ai fait d'abord l'anglais puis j'ai traduit de l'anglais au français.

Tâches accomplies

- Faire le résumé
- Faire l'abstract
- Rendre le résumé et l'abstract

14.05

Tâches à faire

- Faire des tests
- Régler les problèmes
- Créer les fonctionnalités visuels

Liens utiles et idées

Problèmes réglés : - Le wrap du texte dans des textbox multiligne - Crédation de fichier (Mauvais nom de table) - Crédation de commentaires (DateTime = mot réservé) - Crédation de checklist (Mauvais nom de table) - Checkbox to ChecklistItem - Changement de la taille de la police de base pour plusieurs fenêtres - Liste de checklistItem non initialisée

Fonctionnalités visuels : - Couleur sur les sprints (Rouge = déjà passé, Bleu = en cours, Gris = à venir) - Augmentation de la taille des User Stories et des Sprints - Limiter les caractères

Autres : - Mise en ordre de la classe DB

Idées : - Suppression de fichier dans l'user story - Visuel des sprints sous forme de GANTT

Tâches accomplies

- Tests effectués
- Problèmes réglés
- Rajout de couleur pour les sprints
- Rajout d'une limitation sur l'interface utilisant
- Augmentation de la hauteur des éléments
- Création de la fenêtre ChecklistItemEdit

15.05

Tâches à faire

- Retoucher le résumé et l'abstract
- Rendre la nouvelle version
- Implémenter la fenêtre ChecklistItemEdit dans le fonctionnement :
 - Accès
 - Modification après fermeture validée

Liens utiles et idées

J'ai d'abord commencé à faire le résumé et l'abstract puis j'ai rendu sur moodle. Cependant, l'heure de rendu était mise à minuit alors qu'elle était annoncée à 17h. J'ai donc envoyé un message à M. Garcia.

J'ai implémenter la fenêtre d'édition de la checklistItemEdit. J'en ai profité pour modifier quelques fonctions redondantes d'évènement en acceptant l'interface EventArgs au lieu d'accepter les évènements de type plus particulier.

Tâches accomplies

- Retoucher le résumé et l'abstract
- Rendre la nouvelle version
- Implémenter la fenêtre ChecklistItemEdit dans le fonctionnement : -- Accès -- Modification après fermeture validée
- Ajout de limitations sur les utilisateurs assignés

18.05

Tâches à faire

- Implémenter l'ajout d'activités
- Refactoriser le code

Liens utiles et idées

Pour implémenter l'ajout des activités, j'ai pensé qu'il serait mieux que rien ne soit lié à l'interface utilisateur. Cependant, j'ai jugé plus intéressant que ce soit le contrôleur qui s'occupe d'appeler la création des activités. J'ai donc rajouter dans chaque fonction changeant les User Stories un appel à la fonction rajoutant un enregistrement dans les activités.

Après cela, j'ai refactorisé le code en supprimant les fonction redondantes. J'ai ensuite séparé les fonction d'évènement de chaque vue des autres fonctions plus utiles et j'ai regrouper les fonctions dans le contrôleur par leur but tel que la création, mise à jour ou suppression d'enregistrement à la base de données.

J'ai commencé à vouloir faire des tests unitaires. Cependant, pour définir la base de données j'ai dû changer le fonctionnement de la classe DB. Il faut absolument pouvoir définir une base de données sans interface graphique. C'est pourquoi j'ai créer une propriété qui peut être définie pour ouvrir une base de données sans avoir à utiliser l'interface utilisateur

Demain, je m'occuperais de la création des tests unitaires

Tâches accomplies

- Implémenter l'ajout d'activités
- Refactoriser le code
- Définition des tests unitaires

19.05

Tâches à faire

- Implémenter les tests unitaires

Liens utiles et idées

J'ai eu mon fameux problème de moteur de base de données en voulant lancer les tests. J'ai donc dû forcer l'application à se lancer en 64 bits pour que cela fonctionne.

J'ai décidé de tester uniquement la classe Controller et la classe DB.

Tâches accomplies

- Tous les tests du controller
- Les tests de lecture de la classe DB.

20.05

Tâches à faire

- Finir les test unitaires
- Rendre le drag'n'drop multipoint

Liens utiles et idées

J'ai fini de faire les tests unitaires pour la DB et j'ai corrigé quelques problèmes sur le controller.

Afin de rendre le drag'n'drop multipoint, plusieurs posts stackoverflow (<https://stackoverflow.com/questions/3191084/wpf-4-multi-touch-drag-and-drop> , <https://stackoverflow.com/questions/3044834/drag-and-drop-as-multitouch-event-in-wpf> , <https://stackoverflow.com/questions/8444998/multi-touch-screen-and-wpf-listbox> , et bien d'autres) me disent d'utiliser le "Surface Toolkit for Windows Touch". Je vais donc faire un git push et essayer en installant le surface toolkit.

Résumé de la visioconférence : Faire un flag pour supprimer quelque chose plutôt que de le supprimer totalement

J'ai essayé de trouver comment importer le Surface Toolkit mais après un moment à essayer de trouver sans succès comment l'importer, je suis parti sur une solution personnalisée.

Je garde en mémoire les informations de l'objet à drag'n'drop dans des listes. J'ai également rajouté une petite boîte qui suit le curseur pour avoir un support visuel.

J'ai eu des soucis sur le PreviewMouseDown qui ne s'activait pas. Après des recherches, le post de stackoverflow :

<https://stackoverflow.com/questions/36656709/wpf-touchscreen-events-not-working-properly> m'a permis de découvrir que le problème était dû au fait que WPF s'attend à la réaction d'un stylet et non à réagir rapidement sur une touche d'utilisateur. Pour changer cela, il a fallut que je désactive cette propriété sur le contrôle voulu.

J'ai également rajouté un style au drag and drop en affichant une petite boîte à l'endroit où on bouge la boîte. Le message de confirmation s'affiche maintenant de manière asynchrone pour ne pas perturber le drag'n'drop des autres gens.

Tâches accomplies

- tests unitaires finis
- Drag'n'drop multipoint sur ProjectMenu
- Drag'n'drop stylisé

21.05

Tâches à faire

- Changer le drag'n'drop sur SprintMenu en multipoint
- Retoucher la documentation

Liens utiles et idées

Pour pouvoir changer le drag'n'drop, j'ai refais comme dans la fenêtre ProjectMenu. Pour ce qui est du déplacement, j'ai tout mis dans l'événement de la fenêtre directement. Cependant, la zone qui est détectée par l'UI comme étant la zone du groupbox est uniquement sur les bords. C'est pourquoi j'ai récupéré les limites du groupbox et j'ai vérifié si le drag se situait dans la limite du groupbox.

De plus, relâcher le drag'n'drop avec un autre drag'n'drop actif recharge bien l'emplacement des userStories sans pour autant casser le drag'n'drop actuel.

Pour la documentation, j'ai vérifié que toutes les images possédaient bien une légende et j'ai rajouté les fonctionnalités et le résultat de mes tests unitaires. Il faudra que je mette à jour le diagramme de classes demain.

Tâches accomplies

- Changer le drag'n'drop sur SprintMenu en multipoint
- Retoucher la documentation

22.05

Tâches à faire

- Mettre à jour les figures
- Mettre à jour la doc

Liens utiles et idées

En mettant à jour les différentes figures, il m'est venu à l'idée de commencer l'implémentation dont j'avais parlé avec Mme. Terrier. J'ai donc rajouté un champ "deletedFlag" dans les tables qui me semblaient pertinentes. J'ai donc omis de le mettre dans les tables de liaison. Je n'ai pas encore implémenter l'utilisation de ce flag mais je le ferai Lundi.

J'ai refais une base de données entièrement vierge et je la garde de côté pour ne pas avoir trop de données.

J'ai restructuré le github car il était un peu trop brouillon. J'ai créé un dossier Documentation pour tout ce qui est documentation et illustrations. Un dossier Installation pour la base de données vierge et, qui sera rajouté à la fin, l'executable de l'application. Le dossier Scrum'o'Wall est toujours mon projet C#.

Tâches accomplies

- Mise à jour des figures
- Mise à jour de la doc
- Changement dans la base de données (Rajout de flag pour la suppression)
- Restructuration du github

25.05

Tâches à faire

- Implémenter l'utilisation du flag de suppression (Modification de DB - GET, Liaison et DELETE)

Liens utiles et idées

Le problème principal que je trouve avec l'implémentation de flag contre la suppression dans la base, je dois donc changer les méthodes du controller pour faire des suppressions en cascade. Je vérifie également que les objets existent bien avant d'effectuer les liaisons au lancement de l'application.

Pour ce qui est de l'acquisition des données, j'ai rajouté pour chaque table pertinente une condition me permettant de récupérer uniquement les bonnes valeurs.

Comme il me restait du temps, j'ai commencé à réaliser l'interface utilisateur pour la suppression. J'ai déjà effectué ce qu'il faut sur la fenêtre d'édition des fichiers en rajoutant un bouton supprimer. J'ai trouvé plus pertinent de rajouter un champ "UserStory" contenant l'objet parent du File dans le fichier plutôt que de passer le userStory en paramètre à travers toutes les vues.

J'ai retapé un peu la documentation. Cependant, il y a quelques petites modifications que j'ai préféré ne pas changer. Par exemple: Les faiblesses dans le SWOT sont obligatoires sinon le SWOT ne serait pas complet. Pour les commentaires sur le Modèle de classe (description de la communication) je pense rajouter un schéma afin d'exemplifier plus facilement.

J'ai donc: supprimer le manuel utilisateur, déplacer le manuel d'installation, déplacer le planning

Tâches accomplies

- Implémentation du flag de suppression dans DB
- Implémentation de la suppression des fichiers
- Retouche de la documentation

26.05

Tâches à faire

- Rajouter la suppression sur les vues
- Faire le diagramme de communication vers la base de données
- Faire le point avec Mme. Terrier

Liens utiles et idées

J'ai commencé par faire le diagramme de communication. J'ai choisi de faire un diagramme de séquence. Cela permet de mieux montrer quelles actions provoquent un accès à la base de données.

J'ai ensuite eu une réunion avec Mme Terrier.

Mettre en évidence des points subtiles : - La communication avec Access - Expliquer le WPF - Drag'n'drop

Chapitre sur l'apport personnel : - Préciser ce qui est récupérer et pas

Conclusion : - Personnel de comment vécu le travailler - Points d'améliorations

J'ai finalement travaillé sur l'ajout de suppression dans les vues. J'ai eu le temps de créer tous les boutons sur toutes les vues et de créer les liaisons entre les vues. Il me manque à implémenter les fonctions dans le controller

Tâches accomplies

- Rajouter la suppression sur les vues
- Faire le diagramme de communication vers la base de données
- Faire le point avec Mme. Terrier

27.05

Tâches à faire

- Implémenter les fonctions de suppression dans le controller.

Liens utiles et idées

J'ai implémenter les fonctions en les reliant aux fonctions correspondantes et j'ai créé les fonctions manquantes dans DB. J'ai également changé la structure des classes afin de contenir directement un objet parent (La user story pour le commentaire ou le projet pour le user story)

J'ai également rajouter un attribut "Deleted" dans les vues de modifications pour pouvoir savoir à la fermeture si l'utilisateur a appuyé sur le bouton de confirmation ou le bouton de suppression. En effet, j'avais commencé par utiliser l'attribut DialogResult des fenêtres car il utilise un booléen nullable ce qui, dans ma première idée permettait de renvoyer 3 types de données différentes avec "Vrai,Faux et Null". Cependant, à la fermeture et avec un DialogResult égal à null, les

Tâches accomplies

- Implémenter les fonctions de suppression dans le controller.
- Implémentation des fonctions manquantes dans la DB.
- Modification des vues pour intégrer la suppression.

28.05

Tâches à faire

- Refaire les tests unitaires

- Régler les problèmes de boutons qui s'intèrcliquent entre les vues

Liens utiles et idées

Grâce au site <https://wpf.2000things.com/tag/touchdown/> j'ai découvert l'ordre de lancement des évènements liés au Touch ce qui me permet de déterminé quel événement il faut appeler.

J'ai réglé le problème des cliques non-attendus en utilisant la méthode TouchUp à la place de TouchDown pour les boutons.

Refaire les tests m'a pris bien plus de temps que prévu. Je ne réussirais pas à terminer à temps. J'ai renommé certaines fonction et je les ai retravaillés pour qu'il n'y ait plus d'erreur peu importe le cas. Je dois encore repasser sur les méthodes de la classe Controller.

Tâches accomplies

- Problèmes des boutons régler
- Tests unitaires de la classe DB refait et documentés.

29.05

Tâches à faire

- Finir les tests unitaires pour le Controller

Liens utiles et idées

J'ai commencé directement par changé les tests unitaires du Controller. Ce qui m'a fait perdre beaucoup des méthodes que j'avais avant en passant le nombre de tests de plus d'une trentaine à 17 pour le controller. J'ai également fait attention à ce que les tests ne puissent pas échouer si aucun enregistrement non nécessaire (autre que les type et la priorité) n'est présent dans la base.

Après cela, j'ai mis à jour différents éléments de la documentation tel que les illustrations.

Tâches accomplies

- Finir les tests unitaires pour le Controller et les documenter
- Mise à jour des illustrations de la doc

01.06

Tâches à faire

- Trouver une solution pour le clavier virtuel
- Implémenter les méthodes pour le mindmap et les node

Liens utiles et idées

Alors que je cherchais un moyen de créer un clavier visuel, je suis tombé sur une astuce pour Windows 10 qui permet d'afficher le clavier visuel si aucun clavier n'est connecté sans que l'ordinateur soit en mode tablette dès que le focus sur un champ de type texte est lancé. <https://blog.mzikmund.com/2015/09/how-to-show-touch-keyboard-on-touch-interaction-with-wpf-textboxes/> Pour cela, il faut aller dans les options > Périphériques > Saisie et activer le champ "Afficher le clavier tactile lorsque vous n'êtes pas en mode tablette et qu'aucun clavier n'est connecté".

Cependant, le mode tablette ne peut pas être activer avec plus d'un écran ce qui rend le réglages de cette option obligatoire et qui rend la connexion d'un clavier non recommandée. (<https://answers.microsoft.com/fr-fr/windows/forum/all/le-mode-tablette-est-gris%C3%A9-impossible-de/cff5b45b-e229-48dc-ad16-e85335189b45?auth=1>)

J'ai rajouté un chapitre dans le manuel d'installation pour le clavier virtuel.

Pour le mindmap et les nodes, j'ai d'abord commencé par créer les méthodes dans la classe DB. J'ai ensuite continuer par le Controller et j'ai fait le CRUD dans le test unitaire pour les deux classes. Après avoir fais les tests unitaires et qu'ils soient valide, j'ai changé la documentation pour collé aux résultats.

Tâches accomplies

- Trouver une solution pour le clavier virtuel
- Implémenter les méthodes pour le mindmap et les node
- Création des tests unitaires pour le mindmap et les node

02.06

Tâches à faire

- Rajouter un accès au mindmap depuis le menu du projet

Liens utiles et idées

Pour rajouter l'accès au mindmap, j'ai préféré rajouter une colonne dans la fenêtre du menu du projet. En effet, les mindmaps sont liés aux projets et il me semble

logique que ce soit leur place. Après cela, j'ai créer les liaisons entre les différentes fenêtres afin d'afficher la fenêtre de création de noeud quand il faut, etc...
Après avoir placé les controls correctements sur la fenêtre de projet, j'ai décidé de refaire un passage sur les différentes vues afin de normaliser un peu l'apparence.
Maintenant, les pages pleines contiennent des boutons ronds et les pages pop-up contiennent des boutons carrés.

Tâches accomplies

- Accès au mindmaps créés.
- Création de noeud créé
- Normalisation de l'interface utilisateur
- Evenements de la fenetre du mindmap créés.

03.06

Tâches à faire

- Dessiner les Node
- Faire le chapitre de l'apport personnel au projet
- Visio avec mme Terrier

Liens utiles et idées

Pour dessiner les nodes, je pense faire une arborescence comme les fichiers de repertoires d'une commande tree. Je mettrai donc le noeud "root" en haut à gauche et je vais créer des colonnes pour chaque profondeur.

Reunion avec MMe Terrier Questions: - Pour le rendu c'est quoi un export git ?

Rendre quelque chose de non executable aux experts

Défense à blanc le mercredi 10 juin. Montrer la structure d'un test unitaire, la structure du drag'n'drop multipoint.

Pendant que j'étais entrain d'essayer de créer correctement les node, j'ai remarqué que les bordures du drag'n'drop multipoint restaient parfois sur place lors du relâchement et lorsqu'un user story était ouvert. J'ai réussi à enlever la zone lors du relâchement mais pas quand l'user story s'ouvre. Cependant, j'ai essayé de supprimer l'apparition de la boîte à l'édition d'un user story mais je n'ai pas réussi. Je n'ai pas eu d'autres choix que d'enlever l'aspect visuel du drag'n'drop

Tâches accomplies

- Vision avec mme terrier
- Dessin des Node
- Correction de l'aspect du drag'n'drop multipoint
- Chapitre sur l'apport personnel réalisé

04.06

Tâches à faire

- Corriger des bugs
- Faire un test de l'interface utilisateur

Liens utiles et idées

Afin d'avoir une interface de mindmap plus simple à créer et modifier, j'ai décidé de changer l'utilisation du canvas par une grid. Je crée ensuite des lignes pour chaque node dessiner et je rajoute une colonne à chaque fois que le nombre de colonnes requises est insuffisant.

J'ai ensuite profité d'une opportunité que mes camarades m'ont proposé de leur montrer mon application pour faire un test en live. J'ai grâce à cela découvert des bugs dans la modification des fichiers et l'obtention des commentaires au chargement de l'application.

J'ai corriger ces problèmes et j'ai suivi un conseil qui m'a été donné pour certains aspects. Par exemple, afficher la date et l'heure d'une activité ou attribuer une couleur aux user stories possédant une date limite.

Tâches accomplies

- Corriger les problèmes de dessin des node
- Corriger la modification des fichier
- Corriger l'obtention des commentaires
- Ajout de couleurs aux user stories avec une date limite
- Modifications sur l'affichage de certaines classes

05.06

Tâches à faire

- Commenter les endroits manquants
- Faire le planning final

Liens utiles et idées

N'ayant pas commenter entièrement mon projet, il a fallu que je repasse sur certains fichiers pour rajouter des précisions sur certaines lignes de code. J'ai également mis un entête sur les fichiers n'en possédant pas.

Après cela, j'ai normalisé les variables et les méthodes du projet en mettant les variables en readonly ou en mettant des majuscules au début des fonctions.

J'ai finalement repassé sur le planning avant de m'ateler à des morceaux de documentation que je n'avais pas encore créé. J'ai remplis la section sur le drag-and-drop

Tâches accomplies

- Commenter les endroits manquants
- Faire le planning final
- Rempli le glisser-deposer dans la doc

06.06

Tâches à faire

- Documentation. chapitre conclusion

Liens utiles et idées

J'ai commencé par les difficultés rencontrés, j'ai continuer sur les améliorations possibles et finalement mon ressenti sur le projet.

Tâches accomplies

- Documentation. chapitre conclusion

07.06 - 08.06

Tâches à faire

- Récupération du code source
- Crédit du manuel utilisateur
- Mettre le journal de bord en pdf
- Rendu du projet

Liens utiles et idées

J'ai commencé par créer le manuel utilisateur. j'ai décidé de faire une description très détaillée des menus principaux puis une description plus brève des différents pop-up. J'ai donc utilisé GIMP afin d'identifier les principales composantes des vues pour les expliquer au mieux.

Tout en faisant le manuel utilisateur, j'ai découvert certains problèmes que j'ai corrigé en même temps.

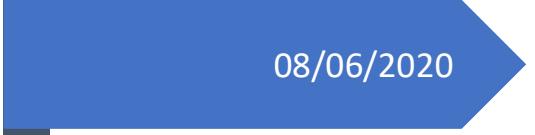
Problèmes trouvés : - Delete checklistItem, checklist, file, - Crédit commentaires - Modification sprint + Etat (Texte)

Après cela, j'ai utilisé les indications de M. Maréchal sur le moodle afin de rendre mon code source en un fichier pdf avec LaTeX.

Finalement, je vais récupérer le journal de bord que voici et le mettre au format pdf afin de le rendre avec la documentation.

Tâches accomplies

- Bug fix
- Récupération du code source
- Crédit du manuel utilisateur
- Journal de bord en pdf + rendu (futur instantané)



08/06/2020

Manuel Utilisateur Scrum'o'Wall

Travail de Diplôme Technicien ES en
informatique.

Session 2019 2020



Travail de Gaël Mariot
Suivi par Anne Terrier
CENTRE DE FORMATION PROFESSIONNEL TECHNIQUE

Table des matières

1	Installation de l'application	3
2	Lancement de l'application	3
3	Fenêtres de menu	4
3.1	Menu principal	4
3.2	Menu du Projet	5
3.3	Menu du Sprint.....	6
3.4	Menu du mindmap	7
3.5	Menu du burndown chart.....	8
4	Fenêtres pop-up.....	9
4.1	Visualisation des activités	9
4.2	Création de checklist	10
4.3	Modification/Suppression de checklist	11
4.4	Création d'objet de checklist	11
4.5	Modification/Suppression d'objet de checklist.....	12
4.6	Gestion des checklists.....	13
4.7	Création de commentaire.....	14
4.8	Gestion des commentaires	15
4.9	Ajout de fichier	16
4.10	Modification/Suppression de fichier	17
4.11	Gestion des fichiers	18
4.12	Création de mindmap	18
4.13	Modification/Suppression de mindmap	19
4.14	Création de nœud.....	19
4.15	Modification/Suppression de nœud.....	20
4.16	Création de projet	20
4.17	Modification/Suppression de projet	21
4.18	Création de sprint.....	22
4.19	Modification/Suppression de sprint.....	22
4.20	Création d'un état	23
4.21	Modification d'un état.....	23
4.22	Gestion des états.....	24
4.23	Création d'un utilisateur.....	24
4.24	Modification/Suppression d'un utilisateur	25

4.25	Gestion des utilisateurs	25
4.26	Création d'une User Story.....	26
4.27	Modification/Suppression d'une User Story.....	27
5	Table des figures	28

1 Installation de l'application

Afin d'installer l'application, il vous faudra aller sur le GitHub du projet puis se rendre dans le dossier « Installation ». (<https://github.com/gaelMRT/Scrum-o-Wall/tree/master/Installation>) Vous trouverez dans ce dossier un exécutable du projet ainsi qu'une base de données vide.

Si l'exécutable ne réussit pas à ouvrir la base de données, installez le moteur de base de données de Access disponible à cette adresse :

<https://www.microsoft.com/fr-FR/download/details.aspx?id=39358>.

2 Lancement de l'application

Au lancement de l'application, une page s'ouvrira afin de déterminer la base de données à utiliser.

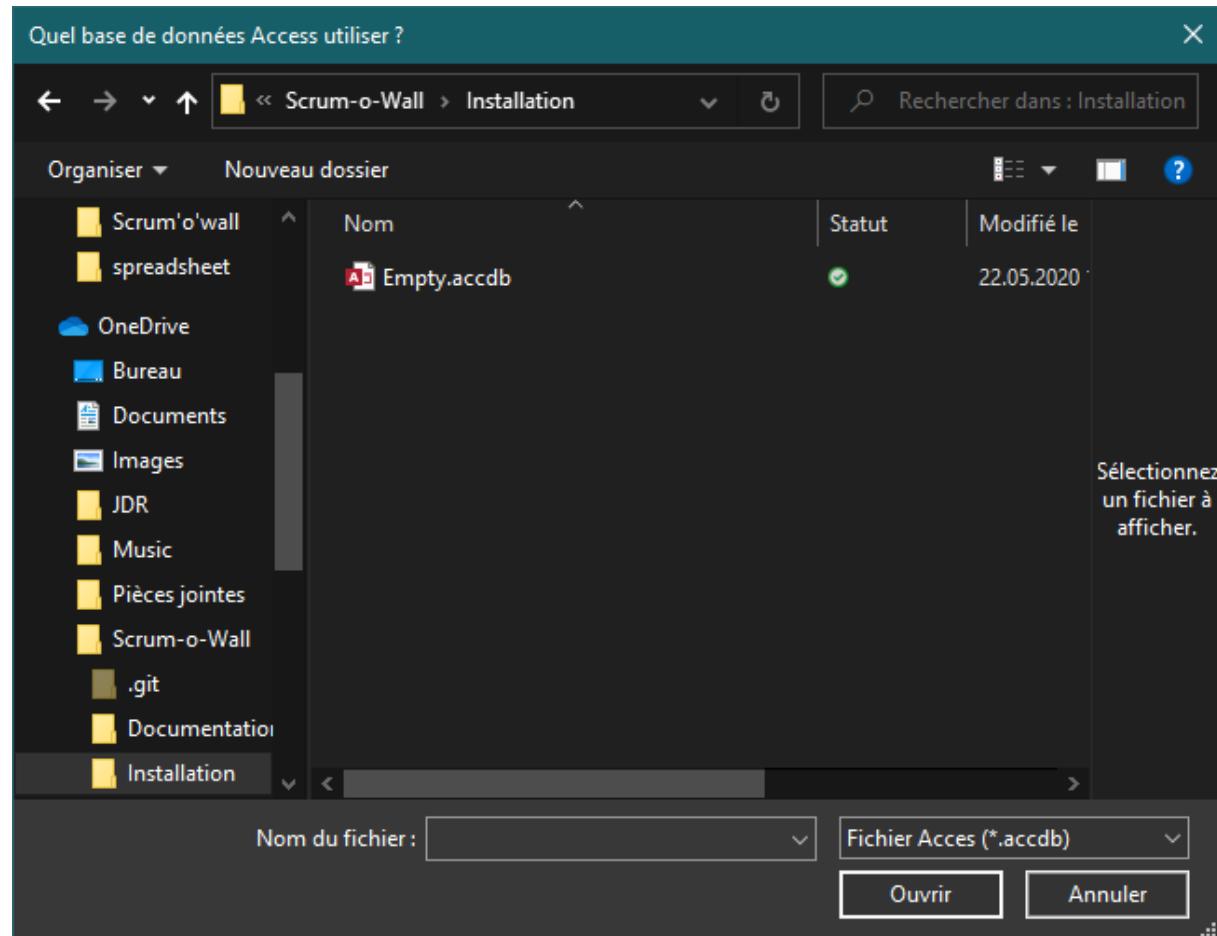


Figure 1 Choix de la base de données

Après avoir choisi la base de données, la fenêtre du menu principal s'ouvrira.

3 Fenêtres de menu

3.1 Menu principal

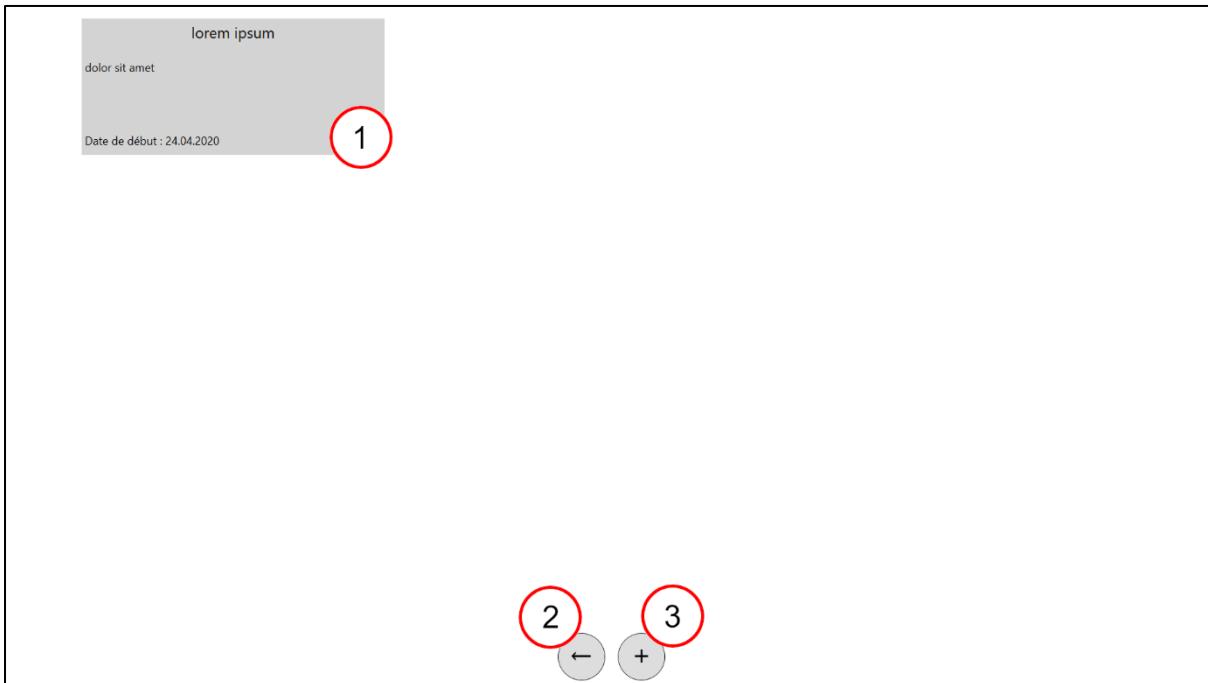


Figure 2 Menu principal

Cette page contient les projets de la base de données. On retrouve 3 éléments importants sur cette page :

1. **L'encadré de projet.** Cet encadré contient les informations de son projet tel que le nom du projet, la description du projet et sa date de début. Appuyer sur cet encadré amène à la fenêtre de projet.
2. **Bouton de retour.** Ce bouton sert à fermer l'application.
3. **Bouton d'ajout de projet.** Ce bouton permet d'accéder à la fenêtre de Création de projet

Le menu contextuel de cette page permet d'ajouter un projet ou de quitter l'application.

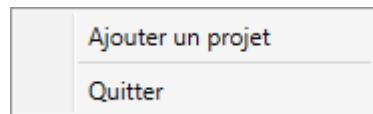


Figure 3 Menu contextuel du menu principal

3.2 Menu du Projet



Figure 4 Menu du projet

Cette page contient les projets de la base de données. On retrouve 7 éléments importants sur cette page :

- Encadré de l'User Story.** Cet encadré contient la description de l'User Story. Il est possible de glisser cet encadré sur un sprint afin de l'attribuer à un sprint et de cliquer dessus afin d'ouvrir la modification de l'User Story. Sa couleur dépend de la date limite de l'User Story. Si elle n'en possède pas, le fond est gris. Si la date est déjà passée, la couleur est rouge. Enfin, si la date n'est pas encore passée, la couleur est bleue.
- Encadré de sprints.** Cet encadré contient les informations d'un sprint. Il est possible de glisser une User Story dessus afin de l'y ajouter et de cliquer dessus afin d'ouvrir la fenêtre de sprint. La couleur du fond dépend de la date du sprint. Si la date du sprint est déjà passé, le sprint est rouge. Si la date du sprint est en cours, la couleur est bleue. Enfin, si la date n'est pas encore dans la portée du sprint, la couleur est grise.
- Encadré de mindmap.** Cet encadré contient le titre du mindmap. Il est possible de cliquer dessus afin d'arriver à la fenêtre de mindmap.
- Bouton d'ajout d'User Story.** Ce bouton permet d'ajouter une User Story
- Bouton d'ajout de sprint.** Ce bouton permet d'ajouter un sprint
- Bouton d'ajout de mindmap.** Ce bouton permet d'ajouter un mindmap
- Bouton de retour.** Ce bouton sert à retourner au menu principal

Le menu contextuel de cette page permet de modifier le projet et de retourner à l'écran précédent.

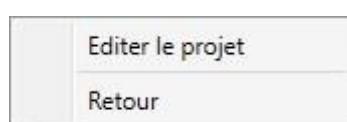


Figure 5 Menu contextuel du menu du projet

3.3 Menu du Sprint

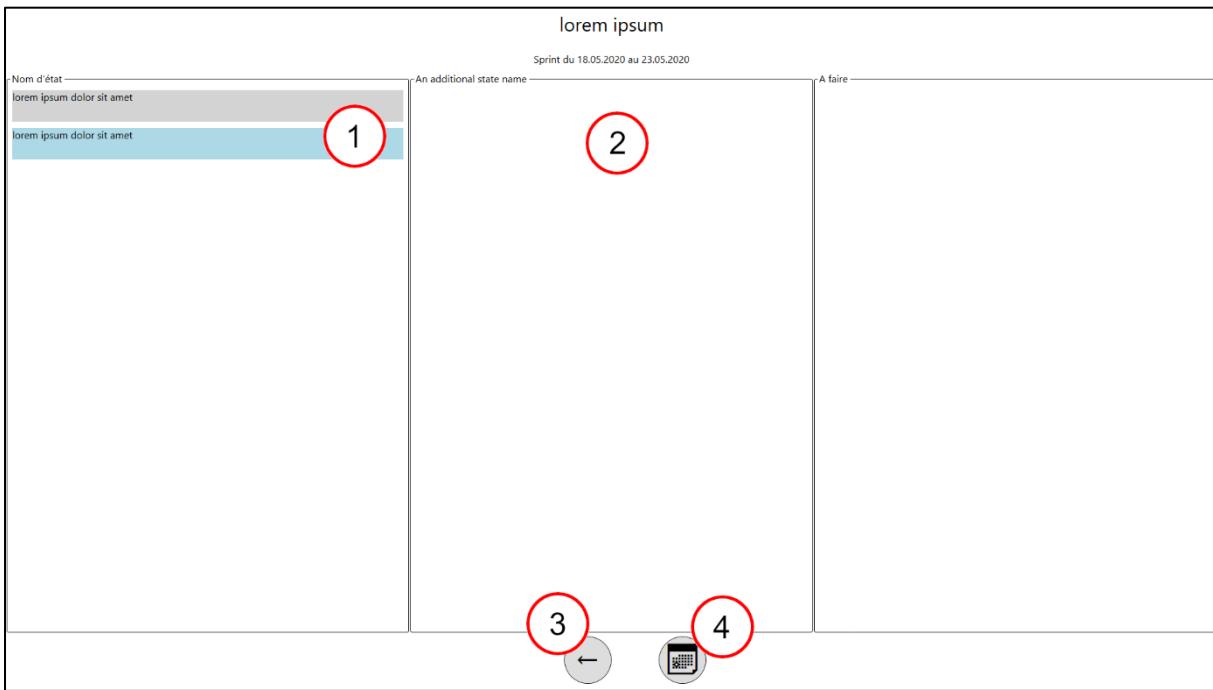


Figure 6 Menu du Sprint

Cette page contient l'aspect du sprint actuel. On retrouve 4 éléments importants sur cette page :

1. **Encadré de l'User Story.** Cet encadré contient la description de l'User Story. Il est possible de glisser cet encadré sur une autre colonne afin de changer son état. Sa couleur dépend de la date de limite de l'user Story. Si elle n'en possède pas, le fond est gris. Si la date est déjà passé, la couleur est rouge. Enfin, si la date n'est pas encore passée, la couleur est bleue point
2. **Colonnes d'état.** Définies dans la modification du projet, ces colonnes permettent d'ordonner son travail avec les User Stories. Glisser et déposer une User Story sur une colonne permet d'assigner l'User Story à cet état.
3. **Bouton de retour.** Ce bouton permet de revenir au menu du projet.
4. **Bouton de burndown chart.** Ce bouton permet d'accéder à la fenêtre du burndown chart de ce sprint.

Le menu contextuel de cette page permet d'ajouter une colonne au projet, d'ouvrir le burndown chart et de modifier le sprint. De plus, une option permettant de retourner au menu de projet est présente.

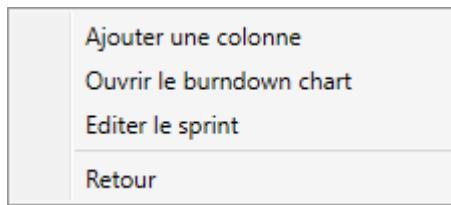


Figure 7 Menu contextuel du menu du sprint

3.4 Menu du mindmap

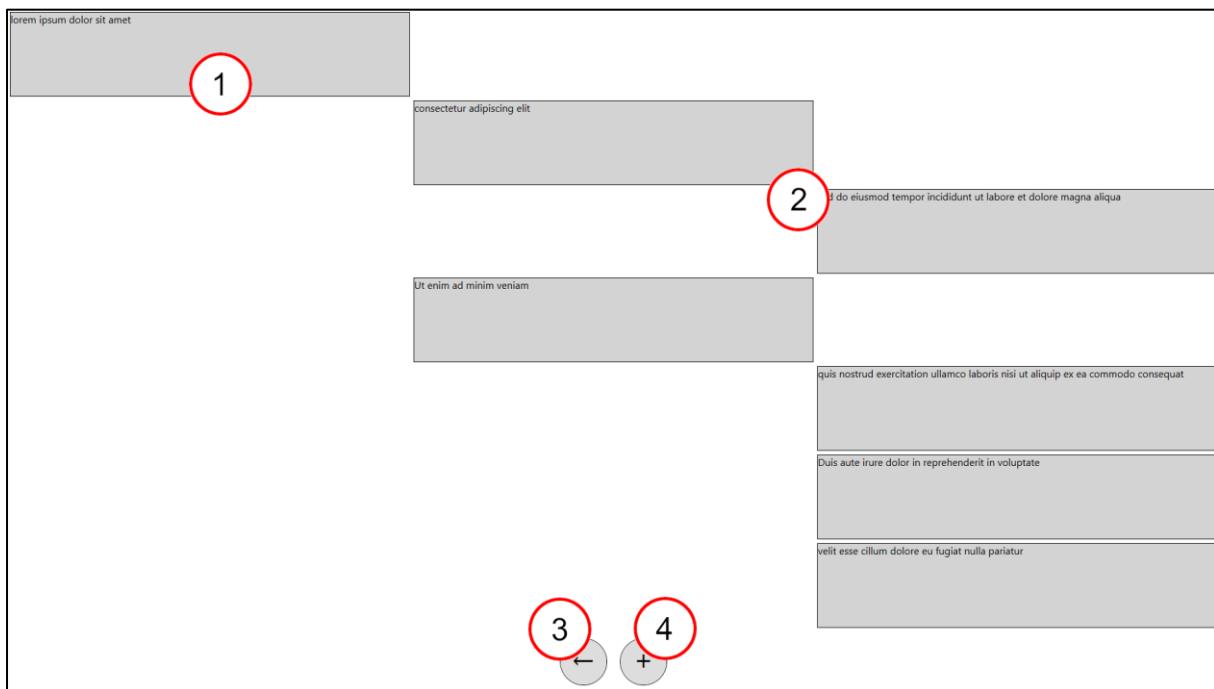


Figure 8 Menu du mindmap

Cette page contient le contenu du mindmap. On retrouve 4 éléments importants sur cette page :

1. **Racine du mindmap.** Ce nœud est la racine du mindmap et contient le texte décrivant le mindmap.
2. **Nœuds inférieurs.** Ces encadrés représentent les enfants directs ou indirects de la racine.
3. **Bouton de retour.** Ce bouton permet de revenir au menu du projet.
4. **Bouton d'ajout de nœud.** Ce bouton permet d'ajouter un nœud au mindmap

Le menu contextuel de cette page donne la possibilité de créer un nœud, modifier un mindmap et retourner au menu précédent.

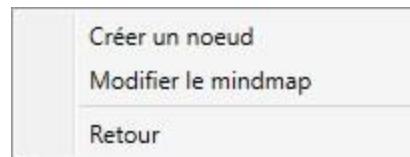


Figure 9 Menu contextuel du mindmap

3.5 Menu du burndown chart

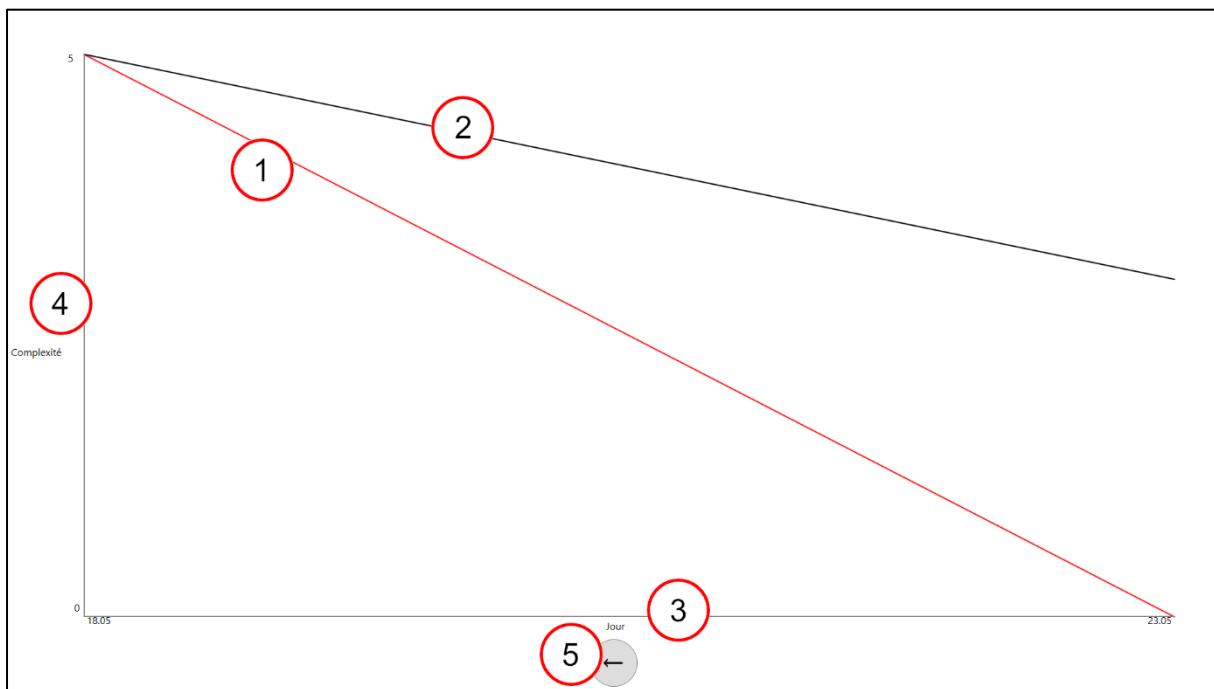


Figure 10 Menu du burndown chart

Cette page contient un calendrier contenant la progression du sprint à travers le temps. On retrouve 5 éléments importants sur cette page :

1. **Ligne de progression idéale.** Cette ligne rouge désigne la progression idéale à avoir durant le sprint pour arriver aux objectifs voulus.
2. **Ligne de progression réelle.** Cette ligne noire désigne la progression actuelle du sprint.
3. **Axe horizontal de durée.** Cet axe représente la portée du sprint de son début à sa fin.
4. **Axe vertical de complexité.** Cet axe représente la valeur de complexité totale des User Stories du sprint en cours.
5. **Bouton de retour.** Ce bouton permet de revenir au menu précédent de l'application.

Le menu contextuel de cette page ne contient qu'une seule possibilité permettant de revenir au menu précédent.

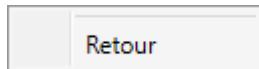


Figure 11 Menu contextuel du burndown chart

4 Fenêtres pop-up

4.1 Visualisation des activités

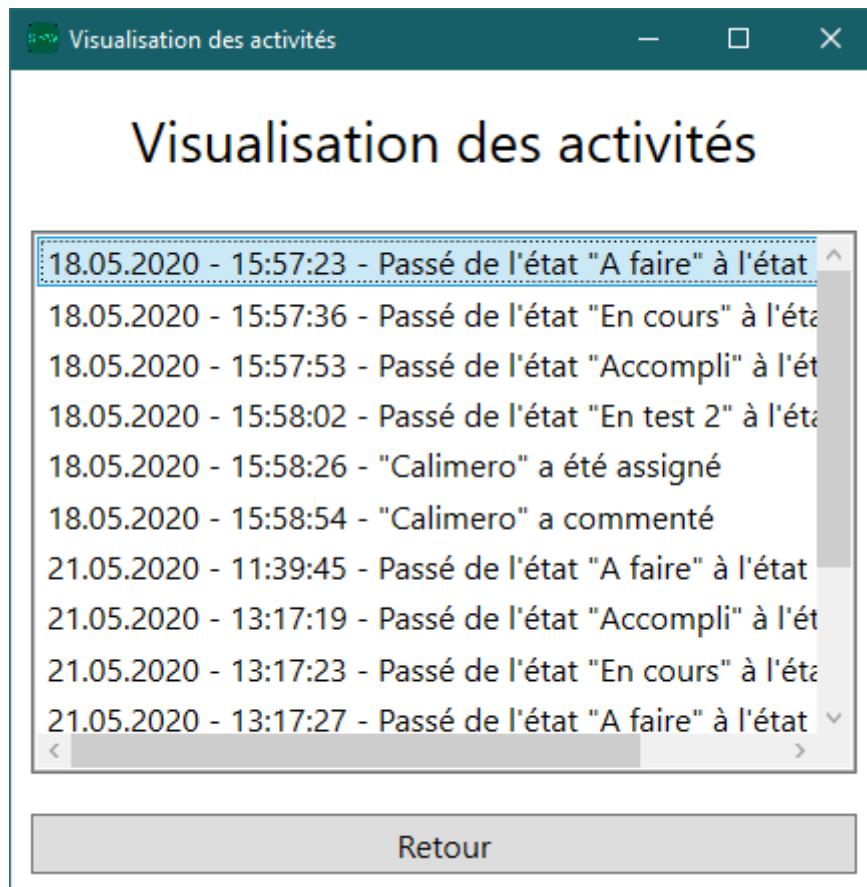


Figure 12 Pop-up de visualisation des activités

Ce pop-up provient du pop-up de Modification/Suppression d'une User Story. Il permet de voir toutes les activités effectuées sur un User Story depuis sa création.

Une activité pouvant avoir un texte très long, il est possible d'ouvrir le Message d'affichage d'activité entière en double cliquant dessus.

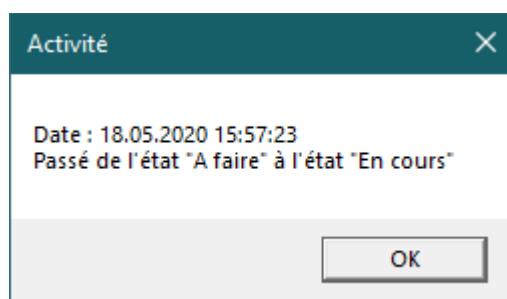


Figure 13 Message d'affichage d'activité entière

4.2 Crédit de checklist

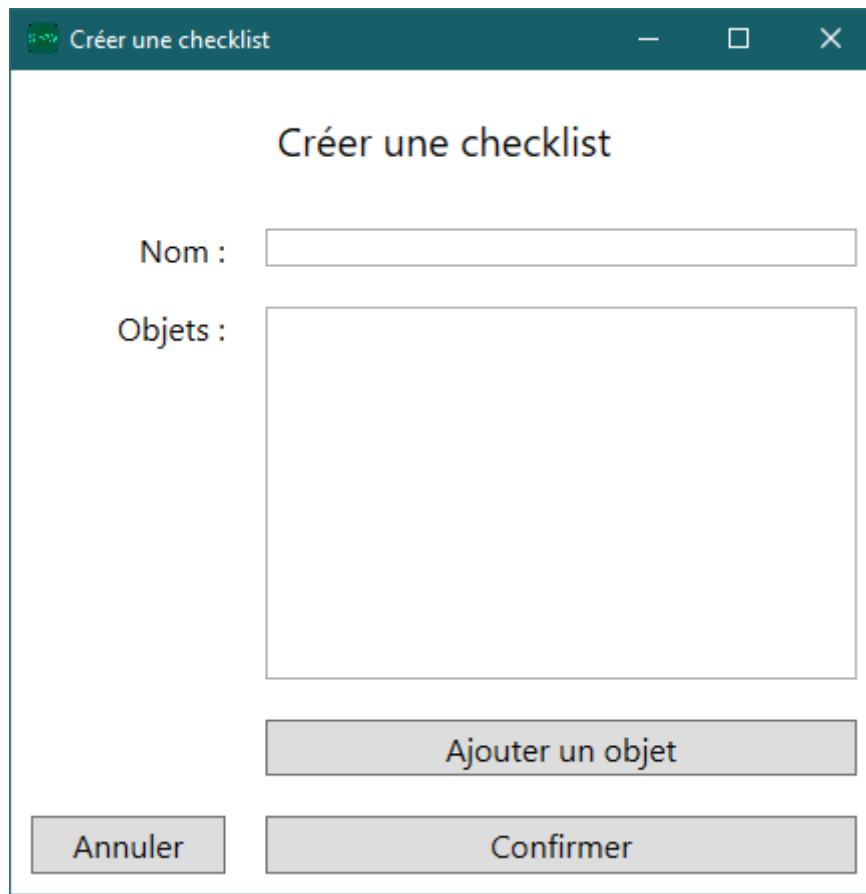


Figure 14 Pop-up de création de checklist

Ce pop-up est appelé depuis la Gestion des checklists.

Il demande un nom et au moins un objet afin d'être confirmé.

4.3 Modification/Suppression de checklist

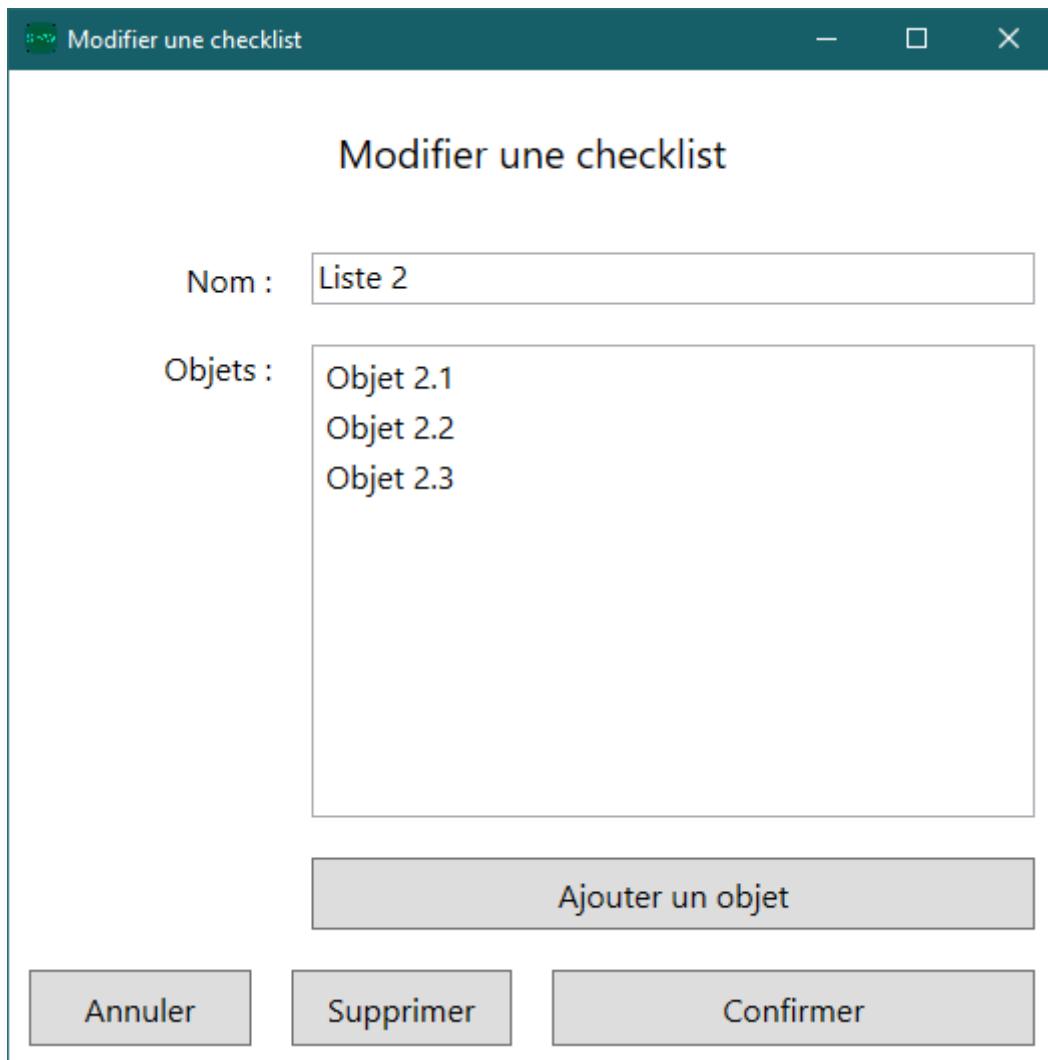


Figure 15 Pop-up de modification de checklist

Ce pop-up est appelé depuis la Gestion des checklists.

Il demande un nom et au moins un objet afin d'être confirmé.

4.4 Crédation d'objet de checklist

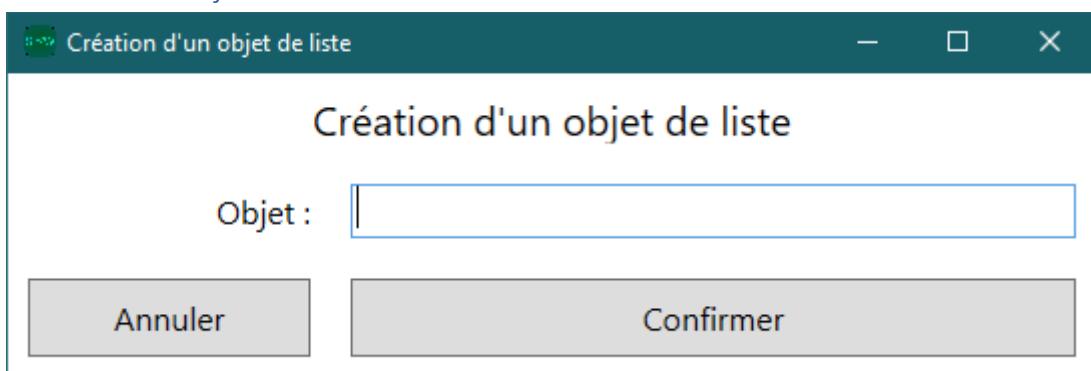


Figure 16 Pop-up de création d'objet de checklist

Ce pop-up est appelé depuis la Modification/Suppression de checklist ou la Crédation de checklist.

Elle demande un nom afin d'être validé.

4.5 Modification/Suppression d'objet de checklist

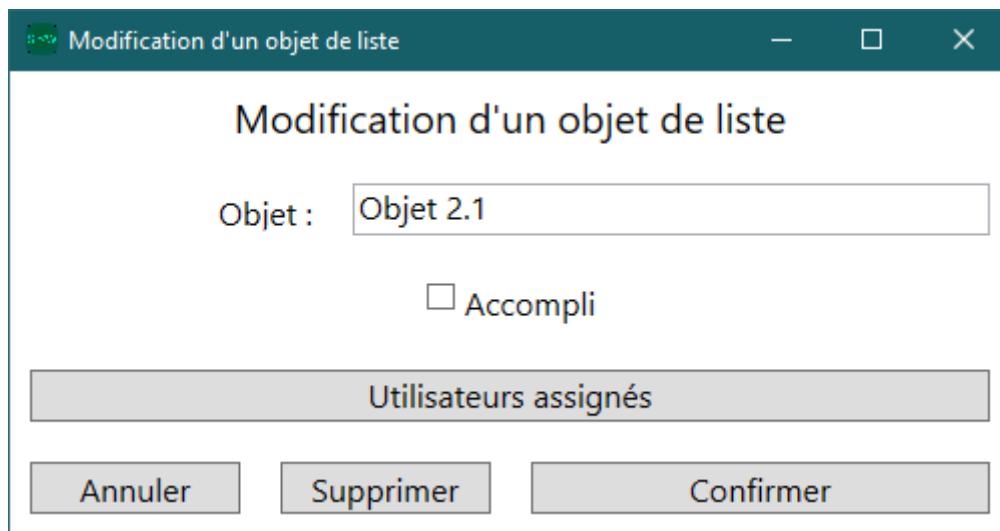


Figure 17 Pop-up de modification d'objet de checklist

Ce pop-up est appelé depuis la Modification/Suppression de checklist ou la Création de checklist.

Elle demande un nom afin d'être validé.

Le bouton « Utilisateurs assignés » ouvre le pop-up de Gestion des utilisateurs afin d'assigner des utilisateurs à cet objet.

4.6 Gestion des checklists

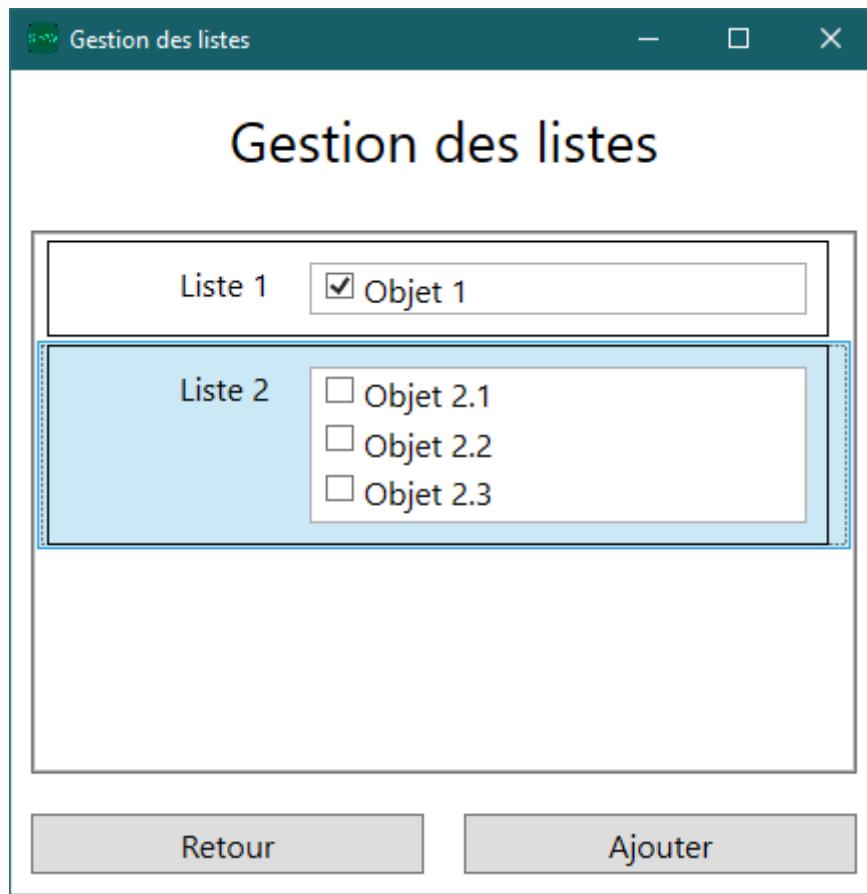


Figure 18 Pop-up de gestion de checklists

Ce pop-up provient du pop-up de Modification/Suppression d'une User Story. Il permet de voir les checklists assignées à une User Story.

Il est possible d'ouvrir la modification d'une checklist en cliquant dessus.

4.7 Crédit de commentaire

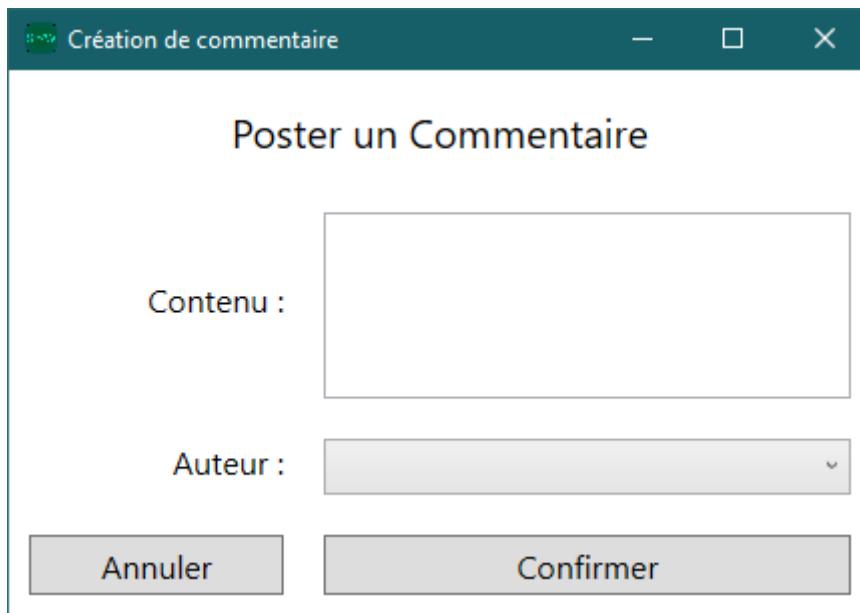


Figure 19 Pop-up de création de commentaire

Ce pop-up est appelé depuis la Gestion des commentaires.

Il requiert du texte dans le champ « Contenu » ainsi qu'un auteur attribué afin d'être confirmé.

Un commentaire ne peut pas être supprimé.

4.8 Gestion des commentaires

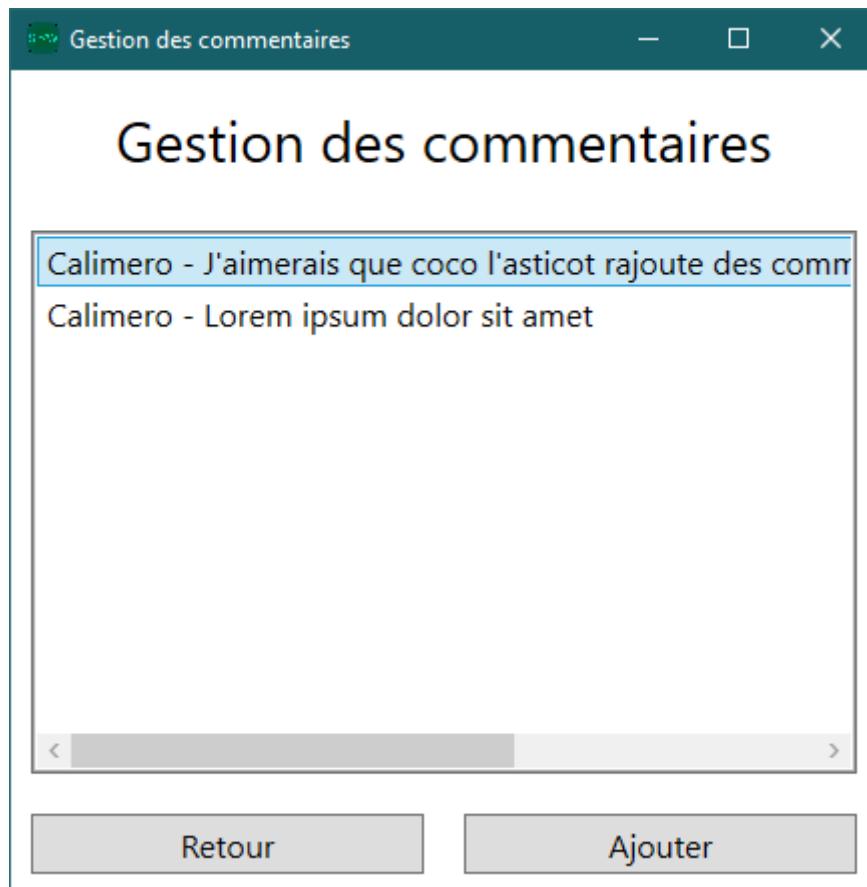


Figure 20 Pop-up de gestion de commentaires

Ce pop-up provient du pop-up de Modification/Suppression d'une User Story. Il permet de voir tous les commentaires postés sur un User Story depuis sa création.

Un commentaire pouvant avoir un texte très long, il est possible d'ouvrir le Message d'affichage de commentaire complet en double cliquant dessus.

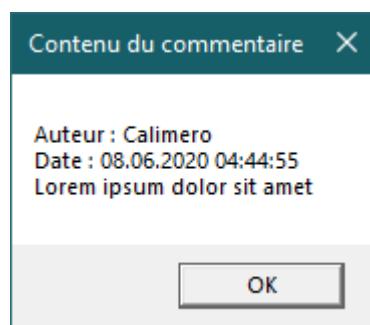


Figure 21 Message d'affichage de commentaire complet

4.9 Ajout de fichier

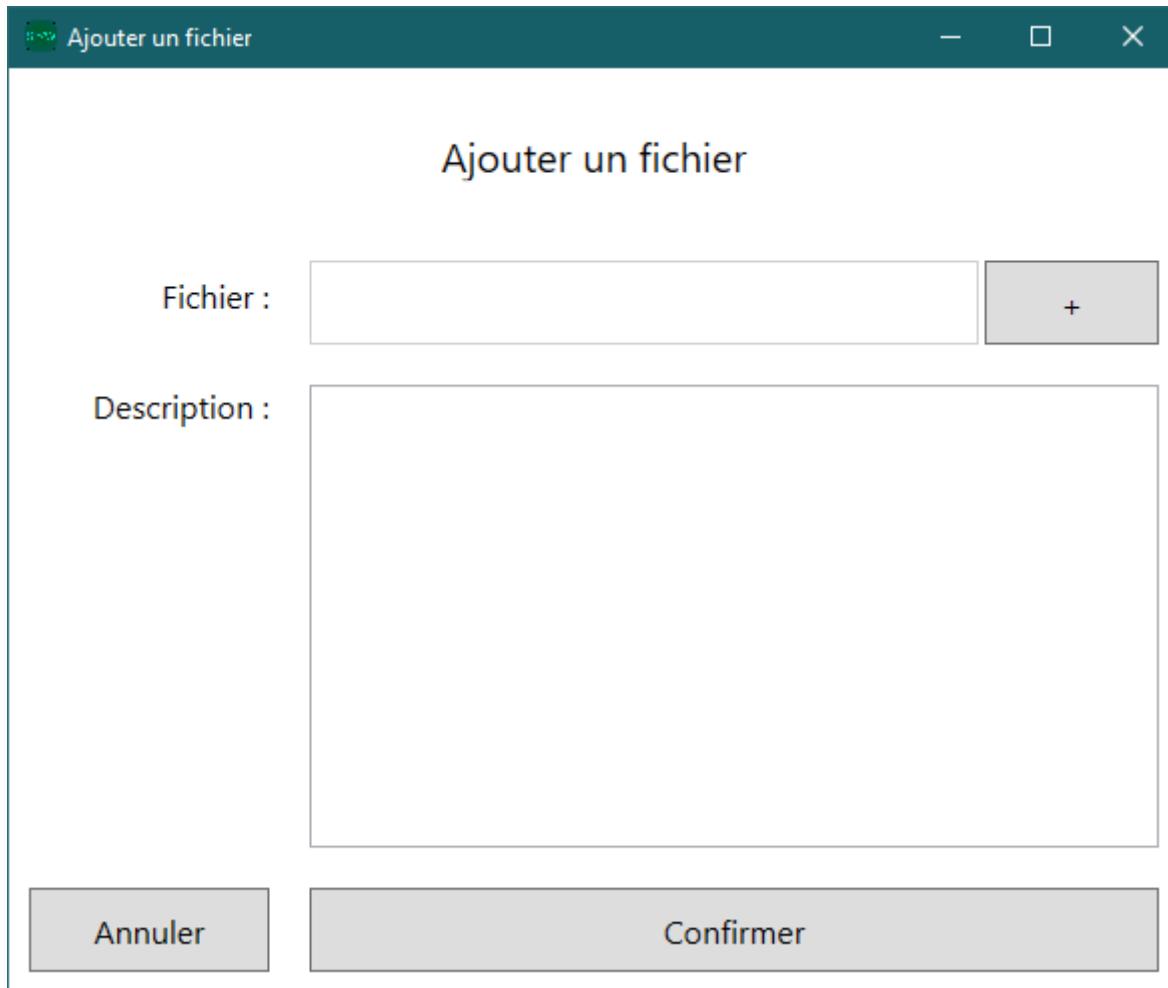


Figure 22 Pop-up d'ajout de fichier

Ce pop-up est appelé depuis la Gestion des fichiers. Il ne peut être confirmé qu'après avoir choisi un fichier et avoir rempli une description.

4.10 Modification/Suppression de fichier

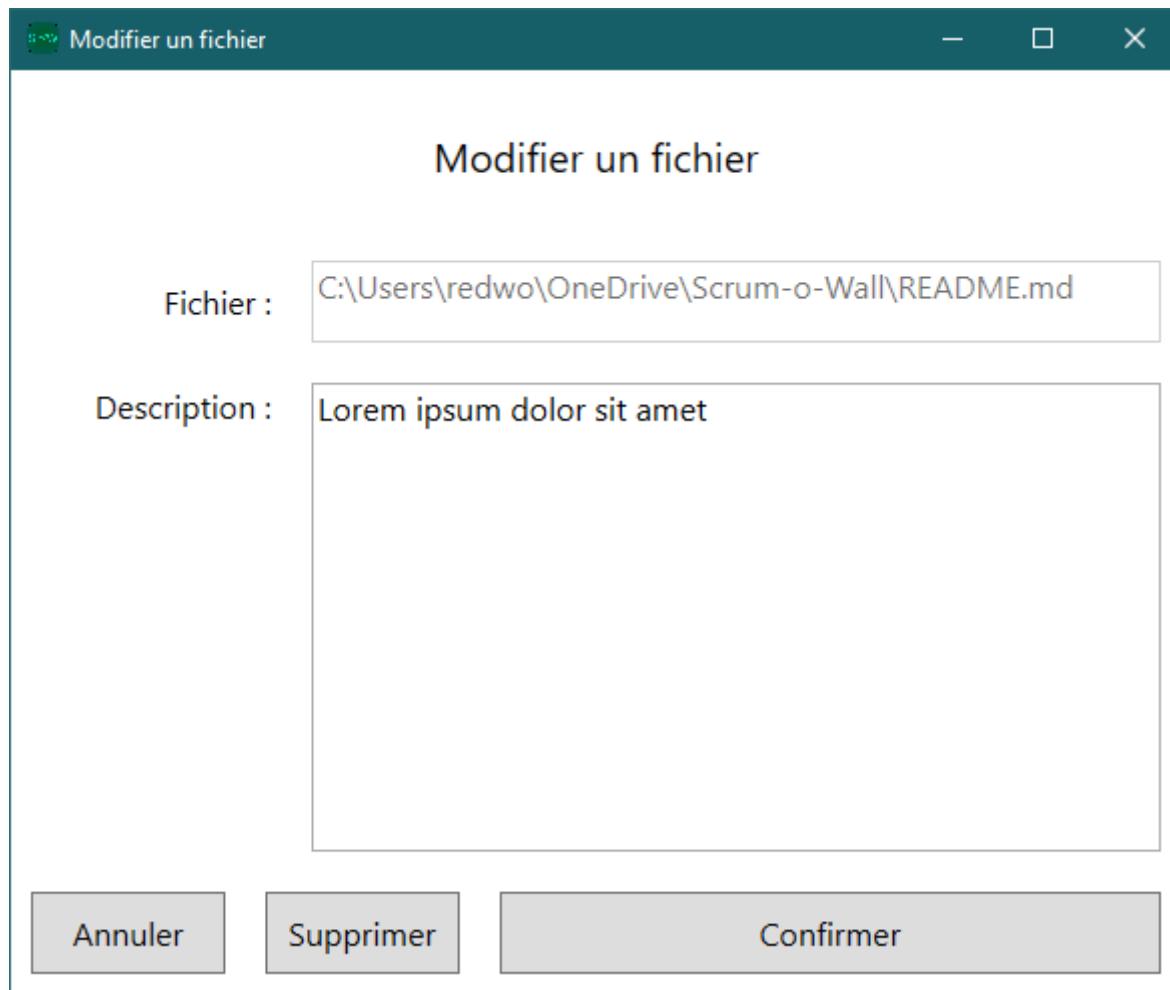


Figure 23 Pop-up de modification de fichier

Ce pop-up est appelé depuis la Gestion des fichiers.

Il n'est pas possible de confirmer la modification avec une description vide. Il n'est pas possible de changer le fichier afin de ne perdre le fichier originel.

4.11 Gestion des fichiers

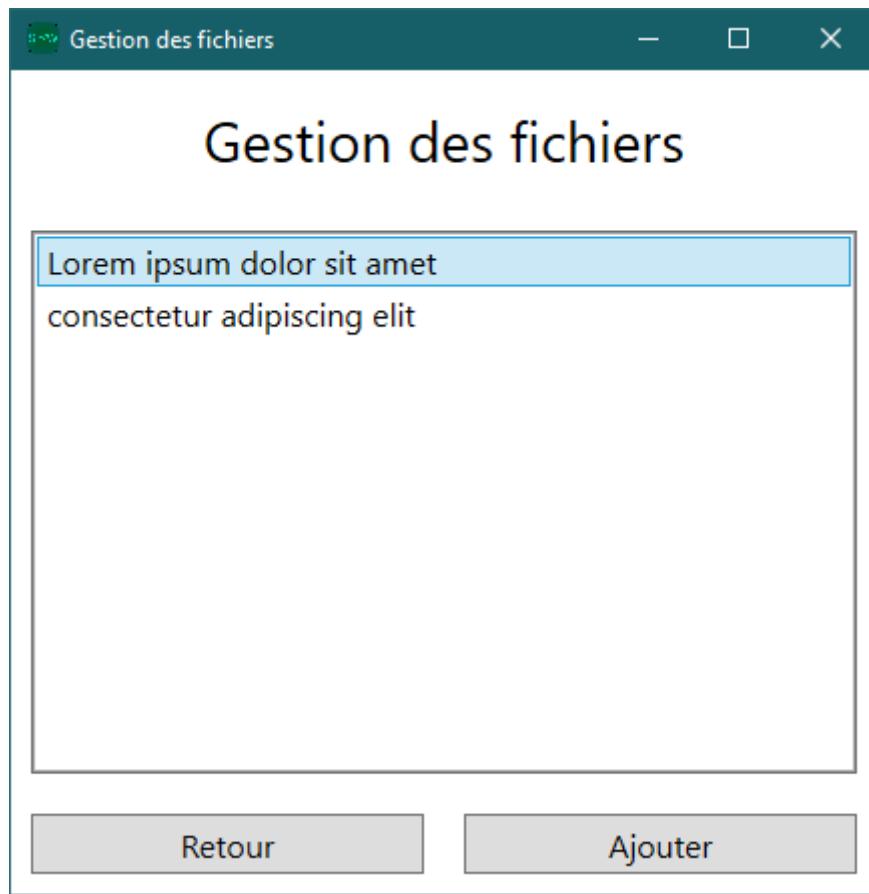


Figure 24 Pop-up de gestion de fichiers

Ce pop-up est appelé par la Modification/Suppression d'une User Story.

Le bouton « Ajouter » permet d'accéder au pop-up d'Add de fichier. Un double clic sur un élément permet d'accéder au pop-up de Modification/Suppression de fichier.

4.12 Création de mindmap

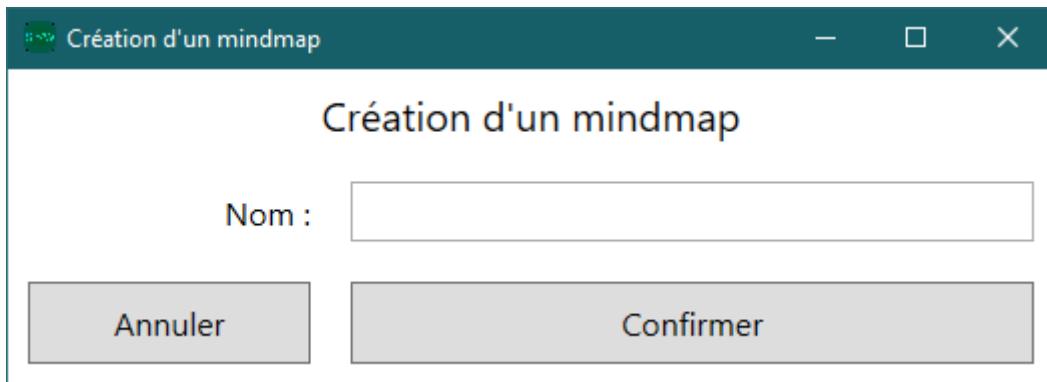


Figure 25 Pop-up de création de mindmap

Ce pop-up est appelé par le Menu du Projet.

Il demande du texte dans le champ « Nom » afin d'accepter une confirmation.

4.13 Modification/Suppression de mindmap

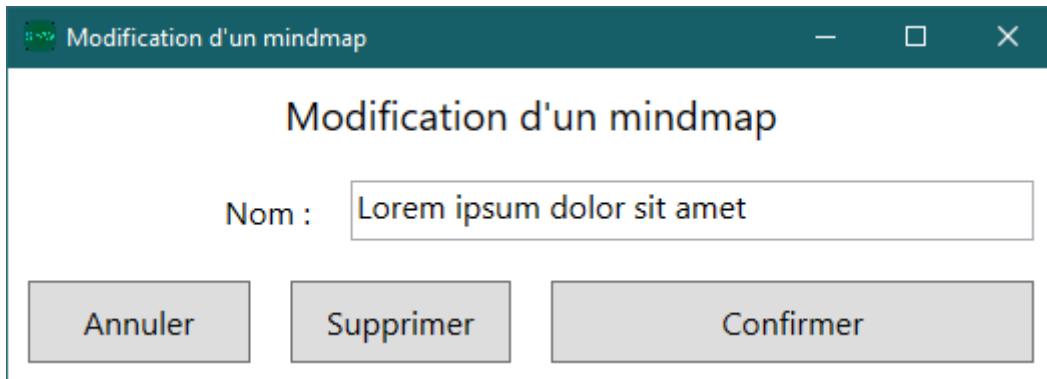


Figure 26 Pop-up de modification de mindmap

Ce pop-up est appelé par le Menu du mindmap.

Il demande du texte dans le champ « Nom » afin d'accepter une confirmation.

4.14 Crédation de noeud

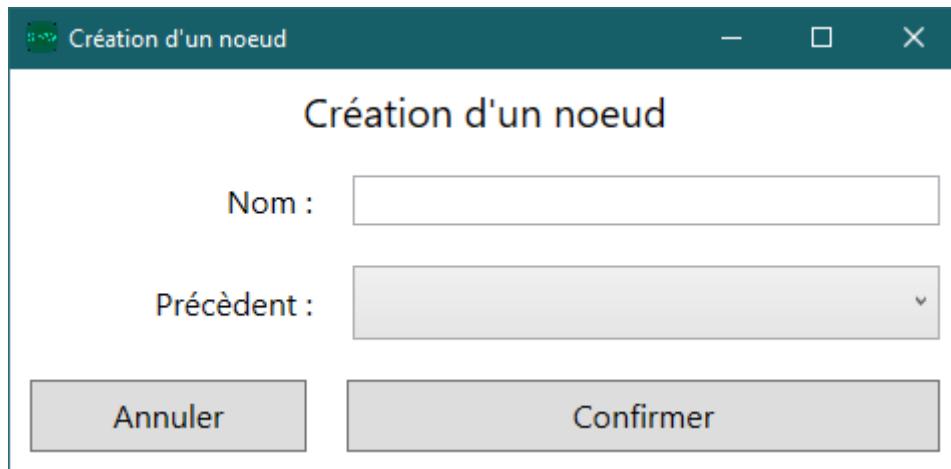


Figure 27 Pop-up de création de noeud

Ce pop-up est appelé par le Menu du mindmap.

Il demande du texte dans le champ « Nom » ainsi qu'un élément sélectionné dans le Précèdent afin d'accepter une confirmation.

4.15 Modification/Suppression de nœud

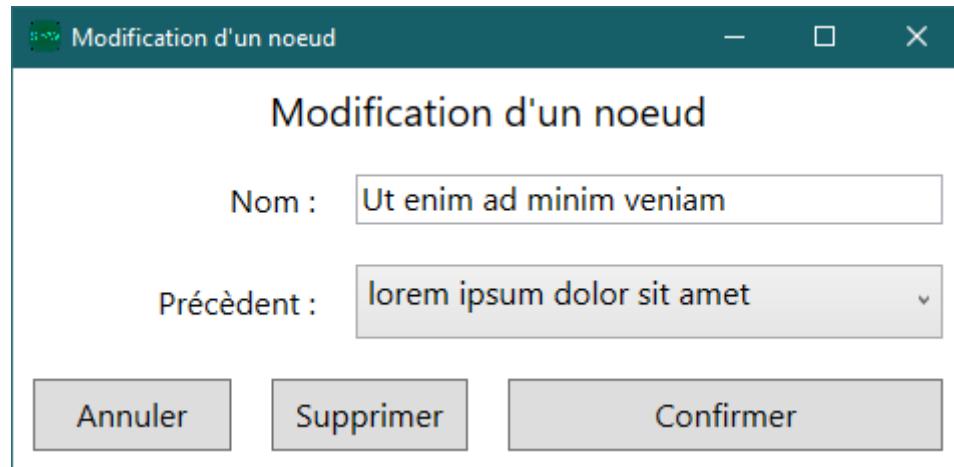


Figure 28 Pop-up de modification de nœud

Ce pop-up est appelé par le Menu du mindmap.

Il demande du texte dans le champ « Nom » ainsi qu'un élément sélectionné dans le Précèdent afin d'accepter une confirmation.

4.16 Crédation de projet

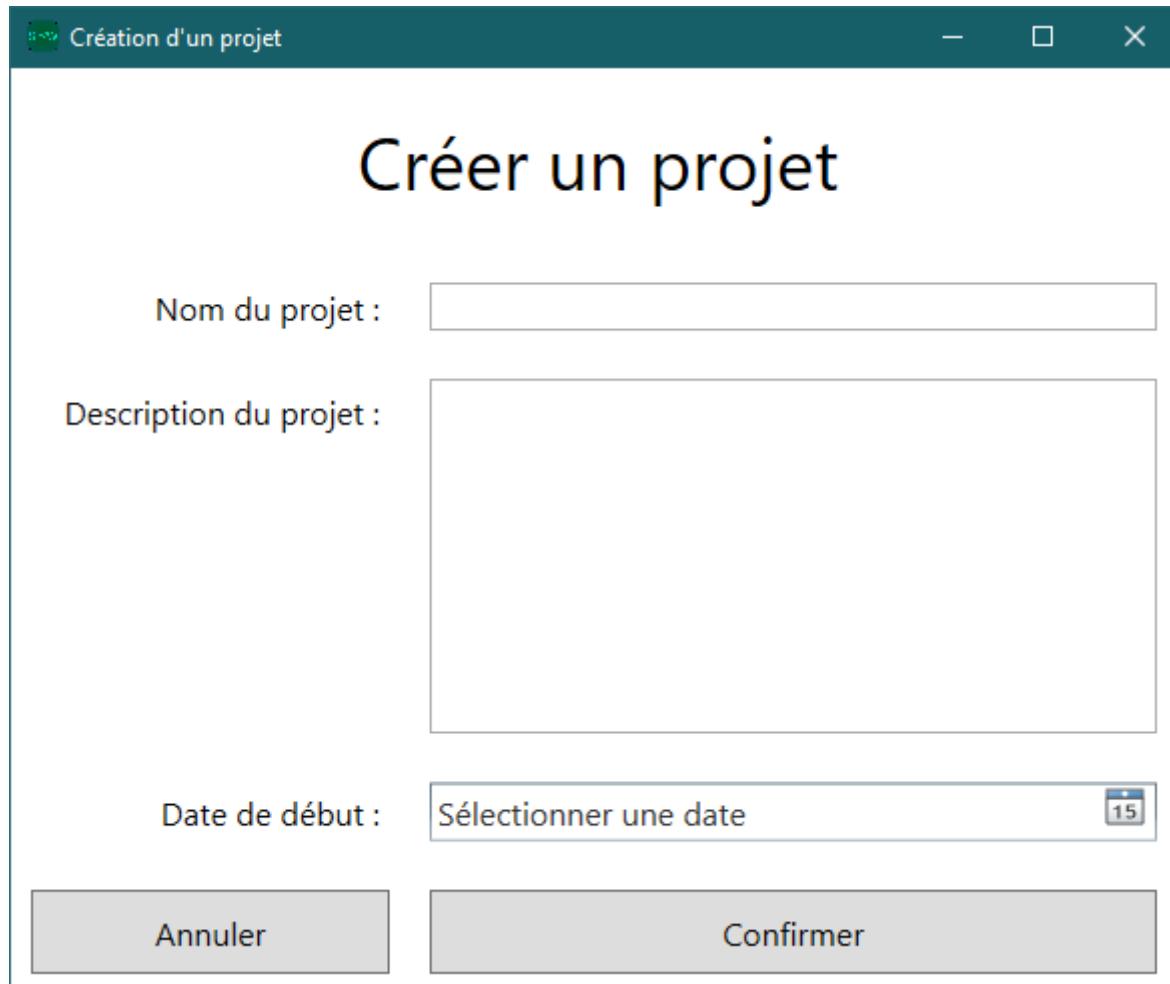


Figure 29 Pop-up de création de projet

Ce pop-up est appelé par le Menu principal.

Il demande du texte dans les champs de nom et de description ainsi qu'une date de début afin d'accepter une confirmation.

4.17 Modification/Suppression de projet

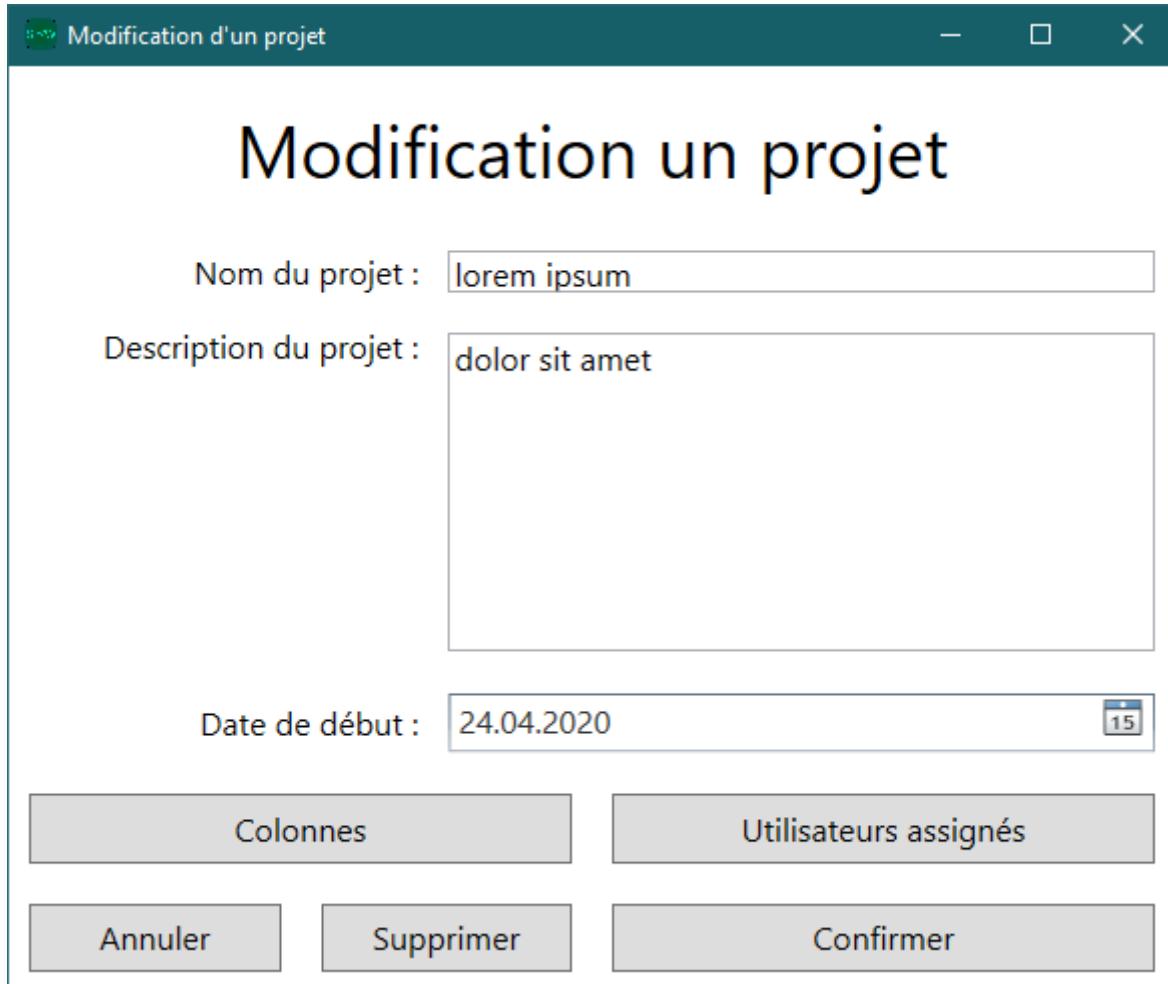


Figure 30 Pop-up de modification de projet

Ce pop-up est appelé par le Menu du Projet.

Il demande du texte dans les champs de nom et de description ainsi qu'une date de début afin d'accepter une confirmation.

Le bouton « Colonnes » renvoie vers le pop-up de Gestion des états et le bouton « Utilisateurs assignés » renvoie vers le pop-up de Gestion des utilisateurs.

Une suppression redirige sur le Menu principal.

4.18 Crédit de sprint

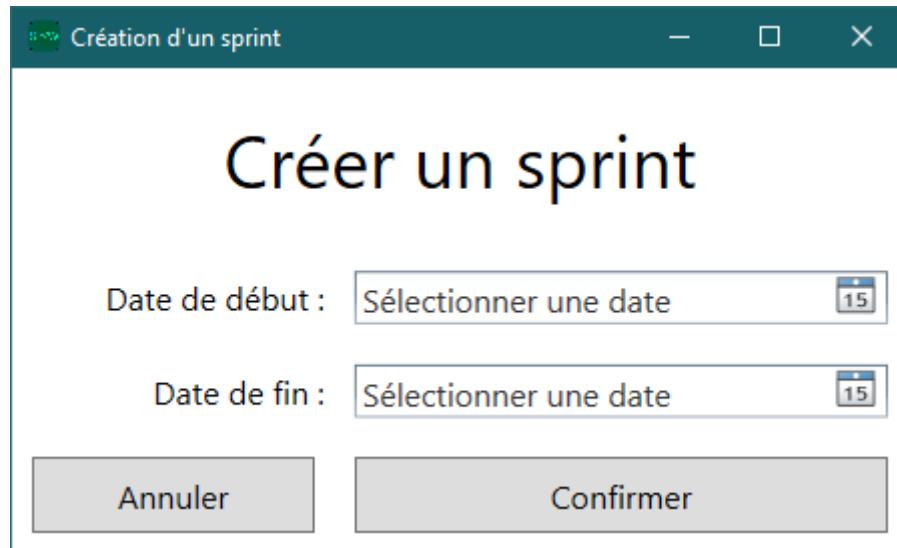


Figure 31 Pop-up de création de sprint

Ce pop-up est appelé depuis le Menu du Projet. Il demande une date de début et une date de fin afin d'être validé.

4.19 Modification/Suppression de sprint

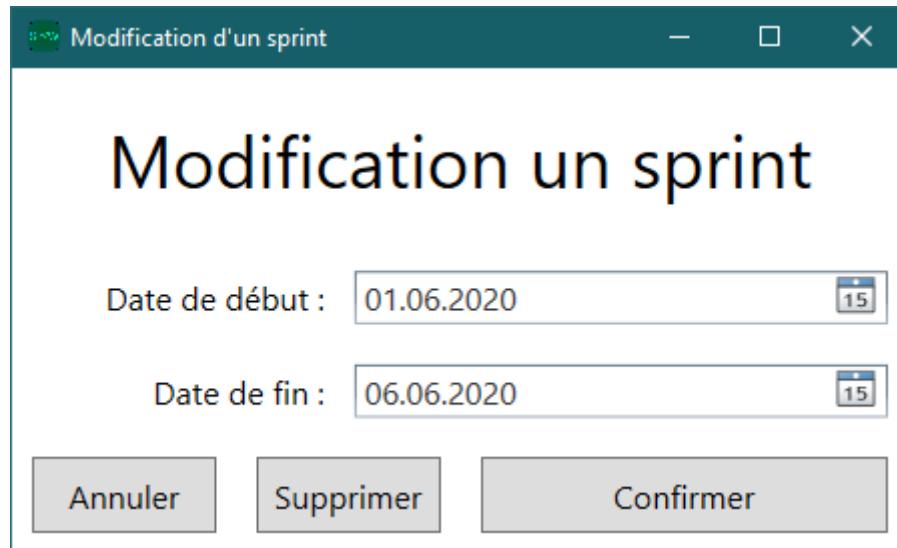


Figure 32 Pop-up de modification de sprint

Ce pop-up est appelé depuis le Menu du Sprint.

Il demande une date de début et une date de fin afin d'être validé.

Une suppression redirige sur le Menu du Projet.

4.20 Crédation d'un état

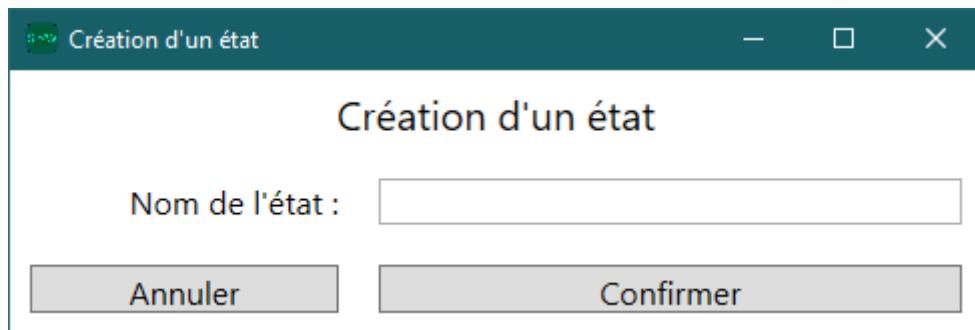


Figure 33 Pop-up de création d'un état

Ce pop-up est appelé depuis la Gestion des états.

Il demande un nom afin d'être validé.

4.21 Modification d'un état

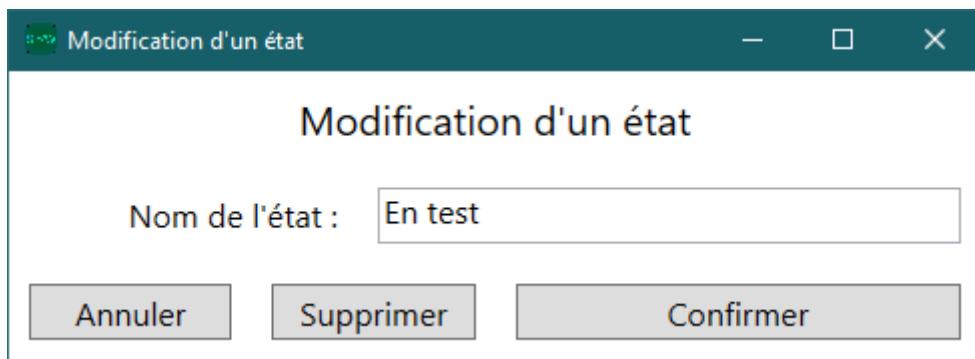


Figure 34 Pop-up de modification d'un état

Ce pop-up est appelé depuis la Gestion des états.

Il demande un nom afin d'être validé.

4.22 Gestion des états

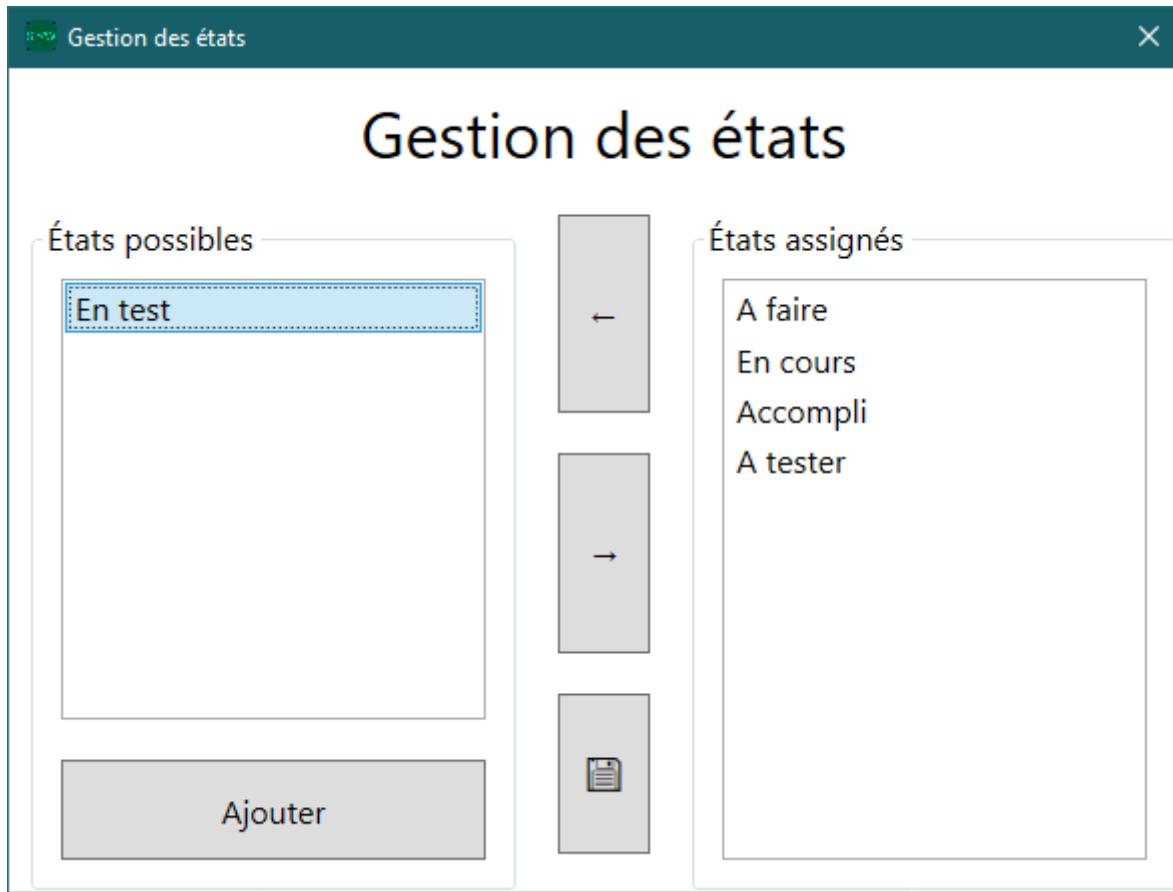


Figure 35 Pop-up de gestion des états

Ce pop-up est appelé depuis la Modification/Suppression de projet ou depuis le Menu du Sprint. Un état doit toujours être assigné.

Le bouton « Ajouter » redirige sur la Création de sprint. Un double clic sur un état redirige sur la Modification d'un état.

Le bouton de sauvegarde enregistre les changements et ferme la fenêtre. Fermer la fenêtre sans appuyer sur ce bouton déclenche un choix d'enregistrement.

4.23 Crédit d'un utilisateur

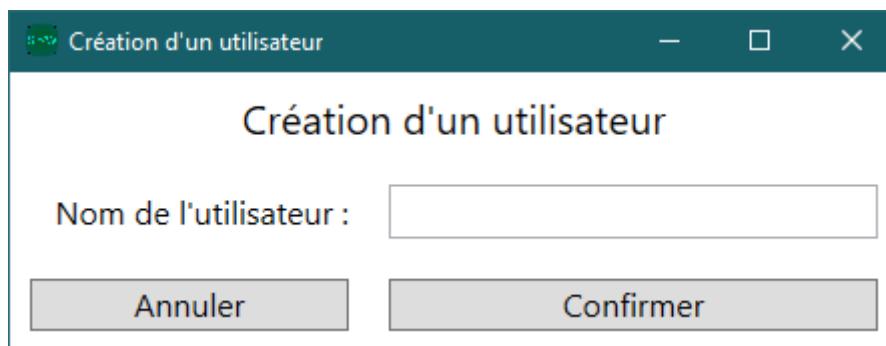


Figure 36 Pop-up de création d'un utilisateur

Ce pop-up est appelé depuis la Gestion des utilisateurs.

Il requiert un nom afin d'être validé.

4.24 Modification/Suppression d'un utilisateur

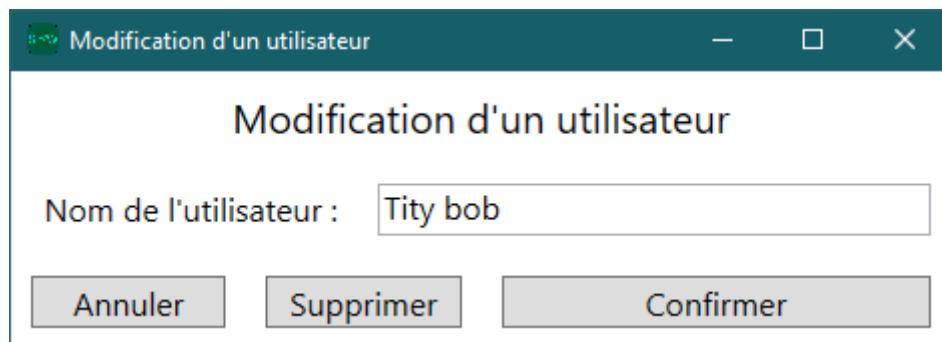


Figure 37 Pop-up de modification d'un utilisateur

Ce pop-up est appelé depuis la Gestion des utilisateurs.

Il requiert un nom afin d'être validé.

4.25 Gestion des utilisateurs

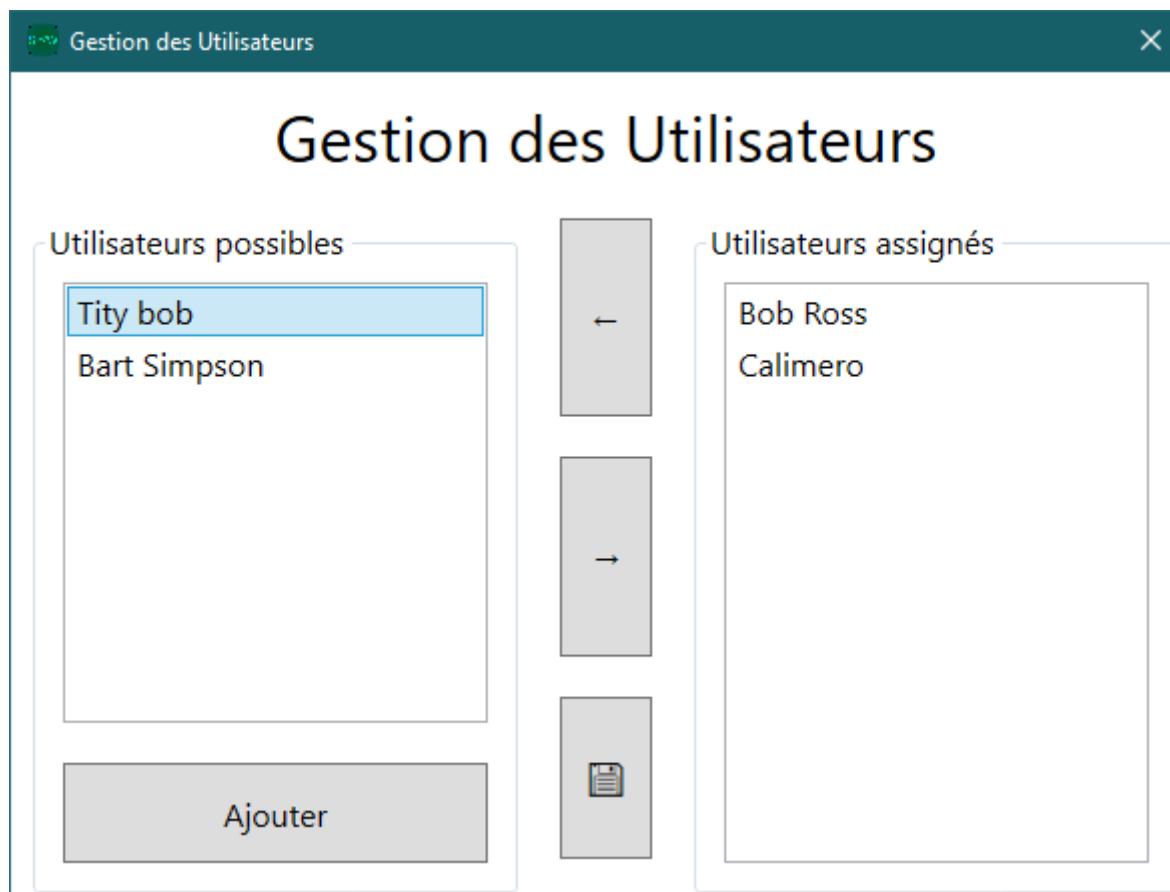


Figure 38 Pop-up de gestion des utilisateurs

Ce pop-up est appelé depuis la Modification/Suppression de projet, la Modification/Suppression d'une User Story ou par la Modification/Suppression d'objet de checklist.

Le bouton « Ajouter » permet d'accéder à la Création d'un utilisateur. Un double clic sur un utilisateur ouvre la Modification/Suppression d'un utilisateur.

Le bouton de sauvegarde enregistre les changements et ferme la fenêtre. Fermer la fenêtre sans appuyer sur ce bouton déclenche un choix d'enregistrement.

4.26 Crédit d'une User Story

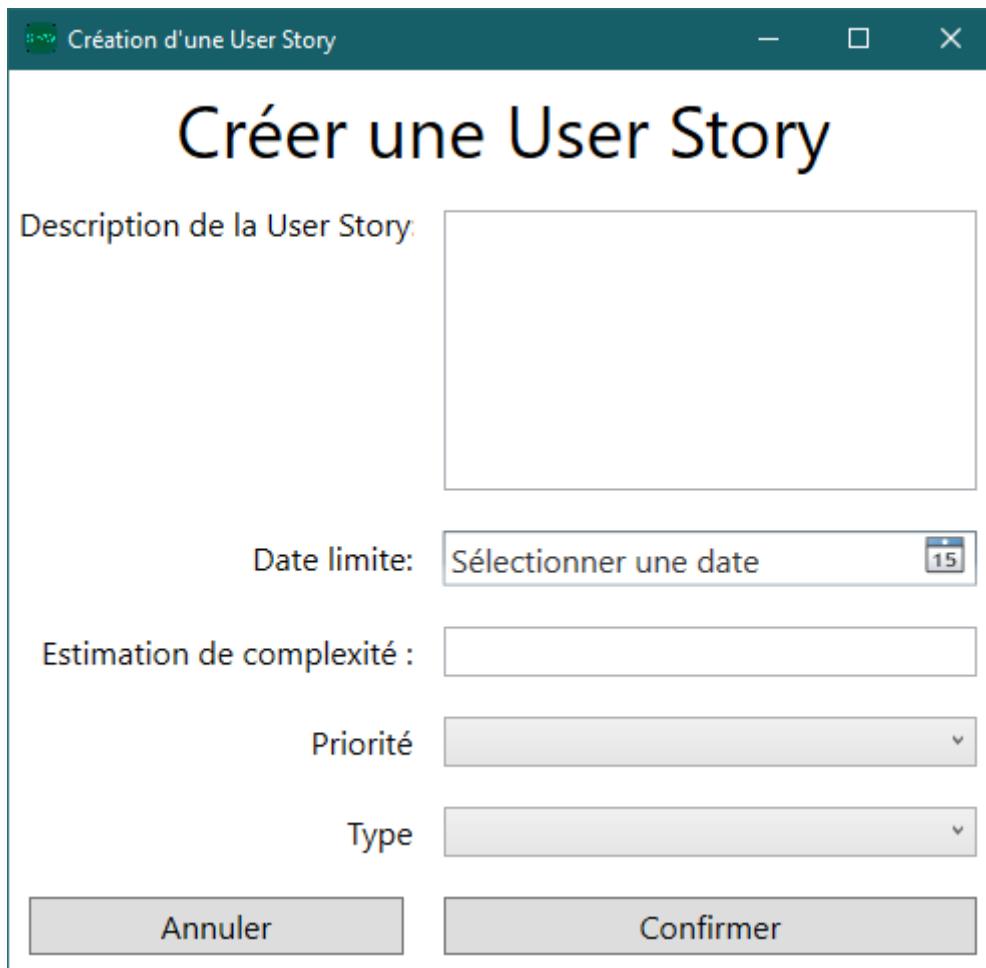


Figure 39 Pop-up de création d'une User Story

Ce pop-up est appelé par le Menu du Sprint ou par le Menu du Projet.

Il requiert une description, une estimation de complexité, une priorité et un type afin d'être validé. La date limite est optionnelle.

4.27 Modification/Suppression d'une User Story

The screenshot shows a Windows-style dialog box titled "Modification d'une User Story". The main area contains several input fields and dropdown menus:

- Description de la User Story :** A text area containing the text: "En tant qu'utilisateur, je veux pouvoir utiliser une application jolie"
- Date limite :** A date picker set to "05.06.2020".
- Estimation de complexité :** A text field containing the value "5".
- Complexité accomplie:** A text field containing the value "2".
- Priorité :** A dropdown menu showing "Moyen".
- Type :** A dropdown menu showing "Tâche".
- Blocké :** An unchecked checkbox.

Below the input fields are several buttons:

- Fichiers
- Commentaires
- Checklists
- Activités
- Utilisateurs assignés
- Annuler
- Supprimer
- Confirmer

Figure 40 Pop-up de modification d'une User Story

Ce pop-up est appelé depuis le Menu du Projet ou le Menu du Sprint.

Il requiert une description, une estimation de complexité, une complexité accomplie (même de zéro), une priorité et un type afin d'être validé. La date limite est optionnelle.

Le bouton de « Fichiers » ouvre la Gestion des fichiers. Le bouton « Commentaires » ouvre la Gestion des commentaires. Le bouton « Checklists » ouvre la Gestion des checklists. Le bouton « Activités » ouvre la Visualisation des activités. Le bouton « Utilisateurs assignés » ouvre la Gestion des utilisateurs.

5 Table des figures

Figure 1 Choix de la base de données.....	3
Figure 2 Menu principal	4
Figure 3 Menu contextuel du menu principal	4
Figure 4 Menu du projet	5
Figure 5 Menu contextuel du menu du projet	5
Figure 6 Menu du Sprint	6
Figure 7 Menu contextuel du menu du sprint.....	6
Figure 8 Menu du mindmap	7
Figure 9 Menu contextuel du mindmap	7
Figure 10 Menu du burndown chart.....	8
Figure 11 Menu contextuel du burndown chart	8
Figure 12 Pop-up de visualisation des activités.....	9
Figure 13 Message d'affichage d'activité entière.....	9
Figure 14 Pop-up de création de checklist	10
Figure 15 Pop-up de modification de checklist	11
Figure 16 Pop-up de création d'objet de checklist	11
Figure 17 Pop-up de modification d'objet de checklist	12
Figure 18 Pop-up de gestion de checklists	13
Figure 19 Pop-up de création de commentaire.....	14
Figure 20 Pop-up de gestion de commentaires.....	15
Figure 21 Message d'affichage de commentaire complet.....	15
Figure 22 Pop-up d'ajout de fichier	16
Figure 23 Pop-up de modification de fichier	17
Figure 24 Pop-up de gestion de fichiers.....	18
Figure 25 Pop-up de création de mindmap.....	18
Figure 26 Pop-up de modification de mindmap.....	19
Figure 27 Pop-up de création de nœud	19
Figure 28 Pop-up de modification de nœud	20
Figure 29 Pop-up de création de projet	20
Figure 30 Pop-up de modification de projet	21
Figure 31 Pop-up de création de sprint	22
Figure 32 Pop-up de modification de sprint.....	22
Figure 33 Pop-up de création d'un état	23
Figure 34 Pop-up de modification d'un état	23
Figure 35 Pop-up de gestion des états.....	24
Figure 36 Pop-up de création d'un utilisateur.....	24
Figure 37 Pop-up de modification d'un utilisateur.....	25
Figure 38 Pop-up de gestion des utilisateurs	25
Figure 39 Pop-up de création d'une User Story	26
Figure 40 Pop-up de modification d'une User Story.....	27

Travail de diplôme 2020 - Scrum'o'wall

Code Source

Gaël Mariot
T.IS-E2A
École d'informatique (CFPT-I)

8 Mai

Table des matières

1 Base	4
1.1 App.xaml	4
1.2 App.xaml.cs	4
1.3 App.config	4
2 Vues	5
2.1 ActivitiesMenu.xaml	5
2.2 ActivitiesMenu.xaml.cs	5
2.3 BurndownChart.xaml	6
2.4 BurndownChart.xaml.cs	7
2.5 ChecklistCreate.xaml	8
2.6 ChecklistCreate.xaml.cs	9
2.7 ChecklistEdit.xaml	9
2.8 ChecklistEdit.xaml.cs	10
2.9 ChecklistItemCreate.xaml	12
2.10 ChecklistItemCreate.xaml.cs	13
2.11 ChecklistItemEdit.xaml	13
2.12 ChecklistItemEdit.xaml.cs	14
2.13 ChecklistMenu.xaml	15
2.14 ChecklistMenu.xaml.cs	16
2.15 CommentCreate.xaml	18
2.16 CommentCreate.xaml.cs	19
2.17 CommentMenu.xaml	19
2.18 CommentMenu.xaml.cs	20
2.19 FileCreate.xaml	21
2.20 FileCreate.xaml.cs	22
2.21 FileEdit.xaml	23
2.22 FileEdit.xaml.cs	23
2.23 FileMenu.xaml	24
2.24 FileMenu.xaml.cs	25
2.25 MainMenu.xaml	27
2.26 MainMenu.xaml.cs	28
2.27 MindmapCreate.xaml	30
2.28 MindmapCreate.xaml.cs	30
2.29 MindmapEdit.xaml	31
2.30 MindmapEdit.xaml.cs	32
2.31 MindmapMenu.xaml	33
2.32 MindmapMenu.xaml.cs	34
2.33 NodeCreate.xaml	36
2.34 NodeCreate.xaml.cs	36
2.35 NodeEdit.xaml	37
2.36 NodeEdit.xaml.cs	38
2.37 ProjectCreate.xaml	39
2.38 ProjectCreate.xaml.cs	40
2.39 ProjectEdit.xaml	40
2.40 ProjectEdit.xaml.cs	41
2.41 ProjectMenu.xaml	42
2.42 ProjectMenu.xaml.cs	44
2.43 SprintCreate.xaml	50
2.44 SprintCreate.xaml.cs	50
2.45 SprintEdit.xaml	51
2.46 SprintEdit.xaml.cs	52
2.47 SprintMenu.xaml	53
2.48 SprintMenu.xaml.cs	53

2.49 StateCreate.xaml	58
2.50 StateCreate.xaml.cs	59
2.51 StateEdit.xaml	59
2.52 StateEdit.xaml.cs	60
2.53 StateMenu.xaml	61
2.54 StateMenu.xaml.cs	62
2.55 UserCreate.xaml	64
2.56 UserCreate.xaml.cs	64
2.57 UserEdit.xaml	65
2.58 UserEdit.xaml.cs	66
2.59 UserMenu.xaml	66
2.60 UserMenu.xaml.cs	67
2.61 UserStoryCreate.xaml	69
2.62 UserStoryCreate.xaml.cs	70
2.63 UserStoryEdit.xaml	71
2.64 UserStoryEdit.xaml.cs	73
3 Modèles	75
3.1 Activity.cs	75
3.2 Checklist.cs	75
3.3 ChecklistItem.cs	76
3.4 Comment.cs	77
3.5 File.cs	77
3.6 IUsersAssigned.cs	78
3.7 MindMap.cs	78
3.8 Node.cs	79
3.9 Priority.cs	80
3.10 Project.cs	81
3.11 Sprint.cs	82
3.12 State.cs	83
3.13 Type.cs	83
3.14 User.cs	84
3.15 UserStory.cs	84
4 Contrôleurs	86
4.1 Controller.cs	86
4.2 DB.cs	97
5 Tests	123
5.1 ControllerTests.cs	123
5.2 DBTests.cs	129

1 Base

1.1 App.xaml

```
1 <Application x:Class="Scrum'o'wall.App"
2     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
3     xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
4     xmlns:local="clr-namespace:Scrum'o'wall"
5     StartupUri="Views/MainMenu.xaml">
6     <Application.Resources>
7
8     </Application.Resources>
9 </Application>
```

Listing 1 – ../../Scrum'o'wall/Scrum'o'wall/App.xaml

1.2 App.xaml.cs

```
1 using System.Windows;
2
3 namespace Scrum'o'wall
4 {
5     /// <summary>
6     /// Logique d'interaction pour App.xaml
7     /// </summary>
8     public partial class App : Application
9     {
10 }
11 }
```

Listing 2 – ../../Scrum'o'wall/Scrum'o'wall/App.xaml.cs

1.3 App.config

```
1 <?xml version="1.0" encoding="utf-8" ?>
2 <configuration>
3     <startup>
4         <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.7.2" />
5     </startup>
6 </configuration>
```

Listing 3 – ../../Scrum'o'wall/Scrum'o'wall/App.config

2 Vues

2.1 ActivitiesMenu.xaml

```
1 <!--
2 * Author      :   Ga l Mariot
3 * Project     :   Scrum'o'wall
4 * File        :   ActivitiesMenu.xaml
5 * Desc.       :   This file contains the basic template of the ActivitiesMenu view
6 -->
7 <Window x:Class="Scrum_o_wall.Views.ActivitiesMenu"
8     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
9     xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
10    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
11    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
12    xmlns:local="clr-namespace:Scrum_o_wall.Views"
13    mc:Ignorable="d"
14    Title="Visualisation des activit s" ←
15        WindowStartupLocation="CenterScreen" Height="450" Width="447.5" ←
16        FontSize="16">
17 <Grid>
18     <Grid.RowDefinitions>
19         <RowDefinition Height="70"/></RowDefinition>
20         <RowDefinition/>
21         <RowDefinition Height="50"/></RowDefinition>
22     </Grid.RowDefinitions>
23     <Grid.ColumnDefinitions>
24         <ColumnDefinition></ColumnDefinition>
25         <ColumnDefinition></ColumnDefinition>
26     </Grid.ColumnDefinitions>
27     <TextBlock Grid.ColumnSpan="2" VerticalAlignment="Center" ←
28         Margin="10,16" TextAlignment="Center" FontSize="28" ←
29         Height="38">Visualisation des activit s </TextBlock>
30     <Border Grid.Row="1" Grid.ColumnSpan="2" Margin="10" ←
31         BorderThickness="1" BorderBrush="{DynamicResource {x:Static SystemColors.GrayTextBrushKey}}">
32         <ListView x:Name="lstActivities" ←
33             MouseDoubleClick="LstActivities_MouseDoubleClick" ←
34             TouchUp="LstActivities_MouseDoubleClick">
35             </ListView>
36         </Border>
37         <Button x:Name="btnCancel" Grid.Row="2" Grid.ColumnSpan="2" Margin="10" ←
38             Click="BtnCancel_Click" TouchUp="BtnCancel_Click">Retour </Button>
39     </Grid>
40 </Window>
```

Listing 4 – ../../Scrum'o'wall/Scrum'o'wall/Views/ActivitiesMenu.xaml

2.2 ActivitiesMenu.xaml.cs

```
1 /*
2 * Author      :   Ga l Serge Mariot
3 * Project     :   Scrum'o'wall
4 * File        :   ActivitiesMenu.xaml.cs
5 * Desc.       :   This file contains the logic in the ActivitiesMenu view
6 */
7 using Scrum_o_wall.Classes;
8 using System;
9 using System.Collections.Generic;
10 using System.Windows;
11 using System.Windows.Controls;
12
13 namespace Scrum_o_wall.Views
14 {
15     /// <summary>
16     /// Logique d'interaction pour ActivitiesMenu.xaml
17     /// </summary>
18     public partial class ActivitiesMenu : Window
19     {
20         private readonly List<Activity> activities;
21         public ActivitiesMenu(List<Activity> someActivities)
22         {
23             activities = someActivities;
24             InitializeComponent();
25             foreach (Activity a in activities)
```

```

26        {
27            lstActivities.Items.Add(a);
28        }
29    }
30
31    private void BtnCancel_Click(object sender, EventArgs e)
32    {
33
34        Close();
35    }
36    private void LstActivities_MouseDoubleClick(object sender, EventArgs e)
37    {
38        ListBox lbx = sender as ListBox;
39        if (lbx.SelectedItem != null)
40        {
41            Activity activity = lbx.SelectedItem as Activity;
42            MessageBox.Show(string.Format("Date : {0}\n{n{1}}", ←
43                activity.DateTime, activity.Description), "Activit ", ←
44                MessageBoxButtons.OK);
45        }
46    }

```

Listing 5 – ../../Scrum'o'wall/Scrum'o'wall/Views/ActivitiesMenu.xaml.cs

2.3 BurndownChart.xaml

```

1 <!--
2 * Author   :   Ga l Mariot
3 * Project  :   Scrum'o'wall
4 * File     :   BurndownChart.xaml
5 * Desc.    :   This file contains the basic template of the BurndownChart view
6 -->
7 <Window x:Class="Scrum_o_wall.Views.BurndownChart"
8     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
9     xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
10    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
11    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
12    xmlns:local="clr-namespace:Scrum_o_wall.Views"
13    mc:Ignorable="d"
14    Title="Scrum'o'Wall" Height="450" Width="800" WindowState="Maximized" ←
15        WindowStyle="None" WindowStartupLocation="CenterScreen" ←
16        FontSize="16" ResizeMode="NoResize">
17
18    <Window.ContextMenu >
19        <ContextMenu >
20
21            <Separator/>
22            <MenuItem Header="Retour" Name="Quit" Click="Quit_Click"/>
23        </ContextMenu>
24    </Window.ContextMenu>
25
26    <Grid>
27        <Line x:Name="lnIdeal" X1="120" Y1="70" StrokeThickness="2" X2="730" ←
28            Y2="330" Stroke="Red"></Line>
29        <Polyline x:Name="lnCurrent" Points="120,70,300,300" ←
30            StrokeThickness="2" Stroke="Black">
31
32        </Polyline>
33        <Border x:Name="brdrGraphic" BorderBrush="Gray" ←
34            BorderThickness="2,0,0,2" Margin="120,70,70,120"/>
35        <Label Content="Complexit " VerticalAlignment="Center"></Label>
36        <Label Content="Jour" VerticalAlignment="Bottom" ←
37            HorizontalAlignment="Center" Margin="383,0,377,91" ←
38            RenderTransformOrigin="0.575,-1.226"></Label>
39        <Label VerticalAlignment="Bottom" Margin="120,0,0,100" ←
40            Name="lblDayBegin">DD.MM</Label>
41        <Label VerticalAlignment="Bottom" Margin="0,0,70,100" ←
42            HorizontalAlignment="Right" Name="lblDayEnd">DD.MM</Label>
43
44        <Label VerticalAlignment="Top" Margin="90,60,0,0" ←
45            HorizontalAlignment="Left" Name="lblComplexityMax">XX</Label>
46        <Label VerticalAlignment="Bottom" Margin="100,0,0,120" ←
47            HorizontalAlignment="Left" Name="lblComplexityZero">0</Label>
48
49        <Button x:Name="btnCancel" Click="Quit_Click" TouchUp="Quit_Click" ←
50            Content="â " VerticalContentAlignment="Top" ←

```

```

37     HorizontalAlignment="Center" Margin="10" Width="75" Height="75" ↵
38     FontSize="45" VerticalAlignment="Bottom">
39         <Button.Resources>
40             <Style TargetType="Border">
41                 <Setter Property="CornerRadius" Value="50"/>
42             </Style>
43         </Button.Resources>
44     </Grid>
45 </Window>

```

Listing 6 – ../../Scrum'o'wall/Scrum'o'wall/Views/BurndownChart.xaml

2.4 BurndownChart.xaml.cs

```

1 /*
2 * Author : Gaël Serge Mariot
3 * Project : Scrum'o'wall
4 * File : BurndownChart.xaml.cs
5 * Desc. : This file contains the logic in the BurndownChart view
6 */
7 using Scrum_o_wall.Classes;
8 using System;
9 using System.Collections.Generic;
10 using System.Windows;
11 using System.Windows.Media;
12
13 namespace Scrum_o_wall.Views
14 {
15     /// <summary>
16     /// Logique d'interaction pour BurndownChart.xaml
17     /// </summary>
18     public partial class BurndownChart : Window
19     {
20         private readonly Sprint sprint;
21         public BurndownChart(Sprint aSprint)
22         {
23             sprint = aSprint;
24             InitializeComponent();
25             Loaded += BurndownChart_Loaded;
26
27             lblDayBegin.Content = sprint.Begin.ToString("dd.MM");
28             lblDayEnd.Content = sprint.End.ToString("dd.MM");
29         }
30
31
32         private void BurndownChart_Loaded(object sender, RoutedEventArgs e)
33         {
34             double daysSinceBegin = (DateTime.Now - sprint.Begin).TotalDays;
35             double totalDays = (sprint.End - sprint.Begin).TotalDays;
36             double elapsedDays = Math.Min(totalDays, daysSinceBegin);
37
38             int totalComplexity = 0;
39             int completedComplexity = 0;
40
41             foreach (KeyValuePair<int, UserStory> keyValuePair in ←
42                     sprint.OrderedUserStories)
43             {
44                 totalComplexity += keyValuePair.Value.ComplexityEstimation;
45                 completedComplexity += keyValuePair.Value.CompletedComplexity;
46             }
47
48             lblComplexityMax.Content = totalComplexity.ToString();
49             double x2Line = elapsedDays / totalDays * (ActualWidth - ←
50                 brdrGraphic.Margin.Left - brdrGraphic.Margin.Right) + ←
51                 brdrGraphic.Margin.Left;
52             double y2Line = completedComplexity / (double)totalComplexity * ←
53                 (ActualHeight - brdrGraphic.Margin.Bottom - ←
54                 brdrGraphic.Margin.Top) + brdrGraphic.Margin.Top;
55
56             PointCollection linePoints = new PointCollection
57             {
58                 new Point(brdrGraphic.Margin.Left, brdrGraphic.Margin.Top),
59                 new Point(x2Line, y2Line)
60             };
61         }
62     }

```

```

58         lnIdeal.X2 = ActualWidth - brdrGraphic.Margin.Right;
59         lnIdeal.Y2 = ActualHeight - brdrGraphic.Margin.Bottom;
60
61         lnCurrent.Points = linePoints;
62     }
63
64     private void Quit_Click(object sender, EventArgs e)
65     {
66
67         Close();
68     }
69
70 }
71 }
```

Listing 7 – ../../Scrum'o'wall/Scrum'o'wall/Views/BurndownChart.xaml.cs

2.5 ChecklistCreate.xaml

```

1 <!--
2 * Author      :   Ga l Mariot
3 * Project     :   Scrum'o'wall
4 * File        :   ChecklistCreate.xaml
5 * Desc.       :   This file contains the basic template of the ChecklistCreate view
6 -->
7 <Window x:Class="Scrum_o_wall.Views.ChecklistCreate"
8     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
9     xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
10    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
11    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
12    xmlns:local="clr-namespace:Scrum_o_wall.Views"
13    mc:Ignorable="d"
14    Title="Cr er une checklist" WindowStartupLocation="CenterScreen" ↵
15        Height="450" Width="447.5" FontSize="16">
16     <Grid>
17         <Grid.RowDefinitions>
18             <RowDefinition Height="70*></RowDefinition>
19             <RowDefinition Height="40*></RowDefinition>
20             <RowDefinition Height="259*></RowDefinition>
21             <RowDefinition Height="50*></RowDefinition>
22         </Grid.RowDefinitions>
23         <Grid.ColumnDefinitions>
24             <ColumnDefinition Width="119*></ColumnDefinition>
25             <ColumnDefinition Width="321*></ColumnDefinition>
26         </Grid.ColumnDefinitions>
27         <TextBlock Grid.ColumnSpan="2" VerticalAlignment="Center" ↵
28             Margin="10,22" TextAlignment="Center" FontSize="20" Height="26" ↵
29             >Cr er une checklist</TextBlock>
30         <TextBlock Grid.Row="1" VerticalAlignment="Center" Margin="10" ↵
31             TextAlignment="Right" Height="20" >Nom :</TextBlock>
32         <TextBlock Grid.Row="2" VerticalAlignment="Top" Margin="10,10,10,0" ↵
33             TextAlignment="Right" Height="21" >Objets :</TextBlock>
34
35         <TextBox Grid.Row="1" MaxLength="255" Grid.Column="1" Margin="10" ↵
36             Name="tbxName" Stylus.IsPressAndHoldEnabled="False"></TextBox>
37         <Grid Grid.Row="2" Grid.Column="1">
38             <Grid.RowDefinitions>
39                 <RowDefinition Height="107*></RowDefinition>
40                 <RowDefinition Height="25*></RowDefinition>
41             </Grid.RowDefinitions>
42             <ListView x:Name="listItems" Margin="10" ↵
43                 Stylus.IsPressAndHoldEnabled="False">
44                 <Button x:Name="btnAddItem" Grid.Row="1" Margin="10" ↵
45                     TouchUp="BtnAddItem_Click" Click="BtnAddItem_Click">Ajouter un ↵
46                     objet</Button>
47             </Grid>
48
49             <Button x:Name="btnCancel" Grid.Row="3" Margin="10" ↵
50                 TouchUp="BtnCancel_Click" Click="BtnCancel_Click">Annuler</Button>
51             <Button x:Name="btnConfirm" Grid.Row="3" Grid.Column="1" Margin="10" ↵
52                 TouchUp="BtnConfirm_Click" Click="BtnConfirm_Click">Confirmer</Button>
53         </Grid>
54     </Window>
```

Listing 8 – ../../Scrum'o'wall/Scrum'o'wall/Views/ChecklistCreate.xaml

2.6 ChecklistCreate.xaml.cs

```
1  /*
2  * Author    :   Gaël Serge Mariot
3  * Project   :   Scrum'o'wall
4  * File      :   ChecklistCreate.xaml.cs
5  * Desc.     :   This file contains the logic in the ChecklistCreate view
6  */
7  using Scrum_o_wall.Classes;
8  using System;
9  using System.Collections.Generic;
10 using System.Windows;
11
12 namespace Scrum_o_wall.Views
13 {
14     /// <summary>
15     /// Logique d'interaction pour ChecklistCreate.xaml
16     /// </summary>
17     public partial class ChecklistCreate : Window
18     {
19         public List<ChecklistItem> itemsToAdd;
20         public ChecklistCreate()
21         {
22             InitializeComponent();
23
24             itemsToAdd = new List<ChecklistItem>();
25
26             Refresh();
27         }
28
29         private void Refresh()
30         {
31             listItems.Items.Clear();
32             foreach (ChecklistItem item in itemsToAdd)
33             {
34                 listItems.Items.Add(item);
35             }
36         }
37
38         private void BtnAddItem_Click(object sender, EventArgs e)
39         {
40             ChecklistItemCreate checklistItemCreate = new ChecklistItemCreate();
41             if (checklistItemCreate.ShowDialog() == true)
42             {
43                 string name = checklistItemCreate.tbxObjet.Text.Trim();
44                 ChecklistItem checklistItem = new ChecklistItem(-1, name, ←
45                                         false, -1);
46
47                 itemsToAdd.Add(checklistItem);
48                 Refresh();
49             }
50         }
51         private void btnCancel_Click(object sender, EventArgs e)
52         {
53             Close();
54         }
55         private void btnConfirm_Click(object sender, EventArgs e)
56         {
57             if (tbxName.Text.Trim().Length > 0 && listItems.Items.Count > 0)
58             {
59                 DialogResult = true;
60                 Close();
61             }
62             else
63             {
64                 MessageBox.Show("Un ou plusieurs champ(s) n'est pas rempli !", ←
65                             "Erreur", MessageBoxButtons.OK, MessageBoxIcon.Error);
66             }
67         }
68     }
69 }
```

Listing 9 – ../../Scrum'o'wall/Scrum'o'wall/Views/ChecklistCreate.xaml.cs

2.7 ChecklistEdit.xaml

```

1 <!--
2 * Author      : Gaël Mariot
3 * Project     : Scrum'o'wall
4 * File        : ChecklistEdit.xaml
5 * Desc.       : This file contains the basic template of the ChecklistEdit view
6 -->
7 <Window x:Class="Scrum_o_wall.Views.ChecklistEdit"
8     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
9     xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
10    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
11    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
12    xmlns:local="clr-namespace:Scrum_o_wall.Views"
13    mc:Ignorable="d"
14    Title="Modifier une checklist" WindowStartupLocation="CenterScreen" ←
15        Height="535.5" Width="537.5" FontSize="16">
16 <Grid>
17     <Grid.RowDefinitions>
18         <RowDefinition Height="70*"/></RowDefinition>
19         <RowDefinition Height="40*"/></RowDefinition>
20         <RowDefinition Height="269*"/></RowDefinition>
21         <RowDefinition Height="50*"/></RowDefinition>
22     </Grid.RowDefinitions>
23     <Grid.ColumnDefinitions>
24         <ColumnDefinition Width="119*"/></ColumnDefinition>
25         <ColumnDefinition Width="321*"/></ColumnDefinition>
26     </Grid.ColumnDefinitions>
27     <TextBlock Grid.ColumnSpan="2" VerticalAlignment="Center" ←
28         Margin="10,21" TextAlignment="Center" FontSize="20" >Modifier une ←
29             checklist</TextBlock>
30     <TextBlock Grid.Row="1" VerticalAlignment="Center" Margin="10,12" ←
31         TextAlignment="Right" >Nom :</TextBlock>
32     <TextBlock Grid.Row="2" VerticalAlignment="Top" Margin="10,10,10,0" ←
33         TextAlignment="Right" >Objets :</TextBlock>
34     <TextBox Grid.Row="1" Grid.Column="1" MaxLength="255" Margin="10" ←
35         Name="txbName" Stylus.IsPressAndHoldEnabled="False"></TextBox>
36     <Grid Grid.Row="2" Grid.Column="1">
37         <Grid.RowDefinitions>
38             <RowDefinition Height="229*"/></RowDefinition>
39             <RowDefinition Height="50*"/></RowDefinition>
40         </Grid.RowDefinitions>
41         <ListView x:Name="listItems" Margin="10" ←
42             Stylus.IsPressAndHoldEnabled="False">
43             </ListView>
44         <Button x:Name="btnAddItem" Grid.Row="1" Margin="10" ←
45             TouchUp="BtnAddItem_Click" Click="BtnAddItem_Click">Ajouter un ←
46             objet</Button>
47     </Grid>
48     <Grid Grid.Row="3" Grid.ColumnSpan="2">
49         <Grid.ColumnDefinitions>
50             <ColumnDefinition></ColumnDefinition>
51             <ColumnDefinition></ColumnDefinition>
52             <ColumnDefinition Width="2*"/></ColumnDefinition>
53         </Grid.ColumnDefinitions>
54         <Button x:Name="btnCancel" Margin="10" Click="BtnCancel_Click" ←
55             TouchUp="BtnCancel_Click">Annuler</Button>
56         <Button x:Name="btnDelete" Grid.Column="1" Margin="10" ←
57             Click="BtnDelete_Click" ←
58             TouchUp="BtnDelete_Click">Supprimer</Button>
59         <Button x:Name="btnConfirm" Grid.Column="2" Margin="10" ←
60             Click="BtnConfirm_Click" ←
61             TouchUp="BtnConfirm_Click">Confirmer</Button>
62     </Grid>
63 </Grid>
64 </Window>

```

Listing 10 – ../../Scrum'o'wall/Scrum'o'wall/Views/ChecklistEdit.xaml

2.8 ChecklistEdit.xaml.cs

```

1 /*
2 * Author      : Gaël Serge Mariot
3 * Project     : Scrum'o'wall
4 * File        : ChecklistEdit.xaml.cs
5 * Desc.       : This file contains the logic in the ChecklistEdit view
6 */

```

```

7  using Scrum_o_wall.Classes;
8  using System;
9  using System.Collections.Generic;
10 using System.Data;
11 using System.Windows;
12 using System.Windows.Controls;
13 using System.Windows.Input;
14
15 namespace Scrum_o_wall.Views
16 {
17     /// <summary>
18     /// Logique d'interaction pour ChecklistEdit.xaml
19     /// </summary>
20     public partial class ChecklistEdit : Window
21     {
22         private readonly Checklist checklist;
23         private readonly Controller controller;
24         private readonly UserStory userStory;
25         private readonly List<ChecklistItem> itemsToAdd;
26         public bool Deleted = false;
27         public ChecklistEdit(Checklist aChecklist, UserStory aUserStory, ←
28             Controller aController)
29         {
30             checklist = aChecklist;
31             controller = aController;
32             userStory = aUserStory;
33
34             InitializeComponent();
35
36             itemsToAdd = new List<ChecklistItem>();
37
38             tbxName.Text = checklist.Name;
39             listItems.MouseDoubleClick += ListItems_MouseDoubleClick;
40
41             Refresh();
42         }
43
44         private void Refresh()
45         {
46             listItems.Items.Clear();
47             foreach (ChecklistItem item in checklist.ChecklistItems)
48             {
49                 listItems.Items.Add(item);
50             }
51             foreach (ChecklistItem item in itemsToAdd)
52             {
53                 listItems.Items.Add(item);
54             }
55         }
56
57         private void ListItems_MouseDoubleClick(object sender, ←
58             MouseEventArgs e)
59         {
60             ListView lstView = sender as ListView;
61             ChecklistItem checklistItem = lstView.SelectedItem as ChecklistItem;
62             if (checklist.ChecklistItems.Contains(checklistItem))
63             {
64                 ChecklistItemEdit checklistItemEdit = new ←
65                     ChecklistItemEdit(checklistItem, userStory, controller);
66                 if (checklistItemEdit.ShowDialog() == true)
67                 {
68                     if (checklistItemEdit.Deleted)
69                     {
70                         controller.Delete(checklistItem);
71                     }
72                     else
73                     {
74                         string objet = checklistItemEdit.tbxObjet.Text.Trim();
75                         bool done = checklistItemEdit.chkboxDone.IsChecked == true;
76                         controller.UpdateCheckListItems(objet, done, ←
77                             checklistItem);
78                     }
79                     Refresh();
80                 }
81             }
82         }
83         private void BtnAddItem_Click(object sender, EventArgs e)
84         {
85             ChecklistItemCreate checklistItemCreate = new ChecklistItemCreate();

```

```

82     if (checklistItemCreate.ShowDialog() == true)
83     {
84         ChecklistItem checklistItem = new ChecklistItem(-1, ←
85             checklistItemCreate.tbxObjet.Text.Trim(), false, -1);
86
87         itemsToAdd.Add(checklistItem);
88         Refresh();
89     }
90     private void BtnCancel_Click(object sender, EventArgs e)
91     {
92         Close();
93     }
94     private void BtnConfirm_Click(object sender, EventArgs e)
95     {
96         if (tbxName.Text.Trim().Length > 0 && listItems.Items.Count > 0)
97         {
98             DialogResult = true;
99             Close();
100        }
101        else
102        {
103            MessageBox.Show("Le nom n'est pas rempli ou il n'y a aucun ←
104                objet dans la liste.", "Erreur", MessageBoxButtons.OK, ←
105                MessageBoxIcon.Error);
106        }
107    }
108    private void BtnDelete_Click(object sender, EventArgs e)
109    {
110        if (MessageBox.Show("La liste sera supprimée.\nVoulez-vous vraiment?", "Attention", MessageBoxButtons.YesNo, MessageBoxIcon.Warning) == DialogResult.Yes)
111        {
112            Deleted = true;
113            DialogResult = true;
114            Close();
115        }
116    }
117 }

```

Listing 11 – .../Scrum'o'wall/Scrum'o'wall/Views/ChecklistEdit.xaml.cs

2.9 ChecklistItemCreate.xaml

```

1 <!--
2 * Author      :   Gaël Mariot
3 * Project     :   Scrum'o'wall
4 * File        :   ChecklistItemCreate.xaml
5 * Desc.       :   This file contains the basic template of the ←
6     ChecklistItemCreate view
7 -->
8 <Window x:Class="Scrum_o_wall.Views.ChecklistItemCreate"
9     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
10    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
11    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
12    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
13    xmlns:local="clr-namespace:Scrum_o_wall.Views"
14    mc:Ignorable="d"
15    Title="Création d'un objet de liste" Height="189.532" Width="558.533" ←
16        WindowStartupLocation="CenterScreen" FontSize="16">
17 <Grid>
18     <Grid.ColumnDefinitions>
19         <ColumnDefinition Width="161" />
20         <ColumnDefinition />
21     </Grid.ColumnDefinitions>
22     <Grid.RowDefinitions>
23         <RowDefinition Height="47*" />
24         <RowDefinition Height="50*" />
25         <RowDefinition Height="62*" />
26     </Grid.RowDefinitions>
27     <TextBlock Grid.ColumnSpan="2" TextAlignment="Center" FontSize="20" ←
28         VerticalAlignment="Center" Margin="10">Création d'un objet de ←
29         liste</TextBlock>

```

```

26     <TextBlock Grid.Row="1" TextAlignment="Right" ↵
27         VerticalAlignment="Center" Margin="10" >Objet :</TextBlock>
28     <TextBox Grid.Row="1" Grid.Column="1" MaxLength="255" Margin="10" ↵
29         Name="tbxObjet" Stylus.IsPressAndHoldEnabled="False"></TextBox>
30     <Button x:Name="btnCancel" Grid.Row="2" Grid.Column="0" Margin="10" ↵
31         Click="BtnCancel_Click" TouchUp="BtnCancel_Click">Annuler</Button>
32     <Button x:Name="btnConfirm" Grid.Row="2" Grid.Column="1" Margin="10" ↵
33         Click="BtnConfirm_Click" TouchUp="BtnConfirm_Click">Confirmer</Button>
34   </Grid>
35 </Window>

```

Listing 12 – ../../Scrum'o'wall/Scrum'o'wall/Views/ChecklistItemCreate.xaml

2.10 ChecklistItemCreate.xaml.cs

```

1 /*
2 * Author      :    Ga l Serge Mariot
3 * Project     :    Scrum'o'wall
4 * File        :    ChecklistItemCreate.xaml.cs
5 * Desc.       :    This file contains the logic in the ChecklistItemCreate view
6 */
7 using System;
8 using System.Windows;
9
10 namespace Scrum_o_wall.Views
11 {
12     /// <summary>
13     /// Logique d'interaction pour ChecklistItemCreate.xaml
14     /// </summary>
15     public partial class ChecklistItemCreate : Window
16     {
17         public ChecklistItemCreate()
18     {
19         InitializeComponent();
20     }
21
22         private void BtnCancel_Click(object sender, EventArgs e)
23     {
24
25             Close();
26     }
27         private void BtnConfirm_Click(object sender, EventArgs e)
28     {
29             if (tbxObjet.Text.Trim().Length > 0)
30             {
31                 DialogResult = true;
32                 Close();
33             }
34             else
35             {
36                 MessageBox.Show("Un ou plusieurs champ(s) n'est pas rempli !", ←
37                     "Erreur", MessageBoxButtons.OK, MessageBoxIcon.Error);
38             }
39         }
40     }

```

Listing 13 – ../../Scrum'o'wall/Scrum'o'wall/Views/ChecklistItemCreate.xaml.cs

2.11 ChecklistItemEdit.xaml

```

1 <!--
2 * Author      :    Ga l Mariot
3 * Project     :    Scrum'o'wall
4 * File        :    ChecklistItemEdit.xaml
5 * Desc.       :    This file contains the basic template of the ChecklistItemEdit ←
6     view
7 -->
8 <Window x:Class="Scrum_o_wall.Views.ChecklistItemEdit"
9     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
10    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
11    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
12    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
13    xmlns:local="clr-namespace:Scrum_o_wall.Views"

```

```

13     mc:Ignorable="d"
14     Title="Modification d'un objet de liste" Height="270.407" ←
15     Width="515.033" WindowStartupLocation="CenterScreen" FontSize="16">
16     <Grid>
17         <Grid.ColumnDefinitions>
18             <ColumnDefinition Width="161"></ColumnDefinition>
19             <ColumnDefinition></ColumnDefinition>
20         </Grid.ColumnDefinitions>
21         <Grid.RowDefinitions>
22             <RowDefinition/>
23             <RowDefinition/>
24             <RowDefinition/>
25             <RowDefinition/>
26             <RowDefinition/>
27         </Grid.RowDefinitions>
28         <TextBlock Grid.ColumnSpan="2" TextAlignment="Center" FontSize="20" ←
29             VerticalAlignment="Center">Modification d'un objet de liste</TextBlock>
30         <TextBlock Grid.Row="1" TextAlignment="Right" ←
31             VerticalAlignment="Center" Margin="10" Height="20" >Objet :</TextBlock>
32         <TextBox Grid.Row="1" Grid.Column="1" MaxLength="255" Margin="10" ←
33             Name="txbObjet" Stylus.IsPressAndHoldEnabled="False"></TextBox>
34
35         <CheckBox Grid.Row="2" Grid.ColumnSpan="2" HorizontalAlignment="Center" ←
36             VerticalAlignment="Center" Name="chkbxDone" ←
37             Stylus.IsPressAndHoldEnabled="False">Accompli</CheckBox>
38
39         <Button Click="BtnAssignedUsers_Click" TouchUp="BtnAssignedUsers_Click" ←
40             Margin="10" Name="btnAssignedUsers" Grid.Row="3" ←
41             Grid.ColumnSpan="2">Utilisateurs assignés</Button>
42
43         <Grid Grid.Row="100" Grid.ColumnSpan="2">
44             <Grid.ColumnDefinitions>
45                 <ColumnDefinition></ColumnDefinition>
46                 <ColumnDefinition></ColumnDefinition>
47                 <ColumnDefinition Width="2*"></ColumnDefinition>
48             </Grid.ColumnDefinitions>
49             <Button x:Name="btnCancel" Margin="10" Click="BtnCancel_Click" ←
50                 TouchUp="BtnCancel_Click">Annuler</Button>
51             <Button x:Name="btnDelete" Grid.Column="1" Margin="10" ←
52                 Click="BtnDelete_Click" ←
53                 TouchUp="BtnDelete_Click">Supprimer</Button>
54             <Button x:Name="btnConfirm" Grid.Column="2" Margin="10" ←
55                 Click="BtnConfirm_Click" ←
56                 TouchUp="BtnConfirm_Click">Confirmer</Button>
57         </Grid>
58     </Grid>
59 </Window>

```

Listing 14 – ../../Scrum'o'wall/Scrum'o'wall/Views/ChecklistItemEdit.xaml

2.12 ChecklistItemEdit.xaml.cs

```

1  /*
2   * Author    : Gaël Serge Mariot
3   * Project   : Scrum'o'wall
4   * File      : ChecklistItemEdit.xaml.cs
5   * Desc.     : This file contains the logic in the ChecklistItemEdit view
6   */
7  using Scrum_o_wall.Classes;
8  using System;
9  using System.Windows;
10
11 namespace Scrum_o_wall.Views
12 {
13     /// <summary>
14     /// Logique d'interaction pour ChecklistItemEdit.xaml
15     /// </summary>
16     public partial class ChecklistItemEdit : Window
17     {
18         private readonly ChecklistItem checklistItem;
19         private readonly Controller controller;
20         private readonly UserStory userStory;
21         public bool Deleted = false;
22
23         public ChecklistItemEdit(ChecklistItem aChecklistItem, UserStory ←
24             aUserStory, Controller aController)
25         {

```

```

25     checklistItem = aChecklistItem;
26     controller = aController;
27     userStory = aUserStory;
28
29     InitializeComponent();
30
31     tbxObjet.Text = checklistItem.NameItem;
32     chkbxDone.IsChecked = checklistItem.Done;
33 }
34
35     private void BtnCancel_Click(object sender, EventArgs e)
36 {
37     Close();
38 }
39
40     private void BtnConfirm_Click(object sender, EventArgs e)
41 {
42     if (tbxObjet.Text.Trim().Length > 0)
43     {
44         DialogResult = true;
45         Close();
46     }
47     else
48     {
49         MessageBox.Show("Un ou plusieurs champ(s) n'est pas rempli !", ←
50                         "Erreur", MessageBoxButtons.OK, MessageBoxIcon.Error);
51     }
52 }
53
54     private void BtnDelete_Click(object sender, EventArgs e)
55 {
56     if (MessageBox.Show("L'objet de la liste sera ↵
57                         supprimé. Voulez-vous r(e) ?", "Attention", ←
58                         MessageBoxButtons.YesNo, MessageBoxIcon.Warning) == ←
59                         MessageBoxResult.Yes)
60     {
61         DialogResult = true;
62         Deleted = true;
63         Close();
64     }
65 }
66
67 }
68 ]

```

Listing 15 – ../../Scrum'o'wall/Scrum'o'wall/Views/ChecklistItemEdit.xaml.cs

2.13 ChecklistMenu.xaml

```

1 <!--
2 * Author : Gaël Mariot
3 * Project : Scrum'o'wall
4 * File : ChecklistMenu.xaml
5 * Desc. : This file contains the basic template of the ChecklistMenu view
6 -->
7 <Window x:Class="Scrum_o_wall.Views.ChecklistMenu"
8     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
9     xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
10    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
11    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
12    xmlns:local="clr-namespace:Scrum_o_wall.Views"
13    mc:Ignorable="d"
14    Title="Gestion des listes" Height="450" Width="447.5" ←
15        WindowStartupLocation="CenterScreen" FontSize="16">
16     <Grid>
17         <Grid.RowDefinitions>
18             <RowDefinition Height="70"/></RowDefinition>
19             <RowDefinition/>
20             <RowDefinition Height="50"/></RowDefinition>
21         </Grid.RowDefinitions>
22         <Grid.ColumnDefinitions>
23             <ColumnDefinition></ColumnDefinition>

```

```

23     <ColumnDefinition></ColumnDefinition>
24   </Grid.ColumnDefinitions>
25   <TextBlock Grid.ColumnSpan="2" VerticalAlignment="Center" ←
26     Margin="10,16" TextAlignment="Center" FontSize="28" ←
27     Height="38">Gestion des listes</TextBlock>
28   <Border Grid.Row="1" Grid.ColumnSpan="2" Margin="10" ←
29     BorderThickness="1" BorderBrush="{DynamicResource {x:Static SystemColors.GrayTextBrushKey}}">
30     <ListView x:Name="lstLists" TouchUp="lstLists_TouchUp" ←
31       MouseDoubleClick="lstLists_TouchUp">
32     </ListView>
33   </Border>
34
35   <Button x:Name="btnCancel" Grid.Row="2" Margin="10" ←
36     Click="BtnCancel_Click" TouchUp="BtnCancel_Click">Retour</Button>
37   <Button x:Name="btnAddList" Grid.Row="2" Grid.Column="1" Margin="10" ←
38     Click="BtnAddList_Click" TouchUp="BtnCancel_Click">Ajouter</Button>
39
40 </Grid>
41 </Window>

```

Listing 16 – ../../Scrum'o'wall/Scrum'o'wall/Views/ChecklistMenu.xaml

2.14 ChecklistMenu.xaml.cs

```

1 /*
2 * Author      : Gaël Serge Mariot
3 * Project    : Scrum'o'wall
4 * File       : ChecklistMenu.xaml.cs
5 * Desc.       : This file contains the logic in the ChecklistMenu view
6 */
7 using Scrum_o_wall.Classes;
8 using System;
9 using System.Collections.Generic;
10 using System.Windows;
11 using System.Windows.Controls;
12 using System.Windows.Media;
13
14 namespace Scrum_o_wall.Views
15 {
16     /// <summary>
17     /// Logique d'interaction pour ChecklistMenu.xaml
18     /// </summary>
19     public partial class ChecklistMenu : Window
20     {
21         private readonly UserStory userStory;
22         private readonly Controller controller;
23         public ChecklistMenu(UserStory aUserStory, Controller aController)
24         {
25             controller = aController;
26             userStory = aUserStory;
27
28             InitializeComponent();
29
30             Refresh();
31         }
32
33         private void Refresh()
34         {
35             lstLists.Items.Clear();
36             foreach (Checklist chckLst in userStory.Checklists)
37             {
38                 ColumnDefinition col1 = new ColumnDefinition
39                 {
40                     Width = new GridLength(30, GridUnitType.Star)
41                 };
42                 ColumnDefinition col2 = new ColumnDefinition
43                 {
44                     Width = new GridLength(67, GridUnitType.Star)
45                 };
46
47                 //Create border
48                 Border border = new Border
49                 {
50                     BorderBrush = Brushes.Black,
51                     BorderThickness = new Thickness(1),
52                     Width = 390,
53                     Tag = chckLst
54                 };
55
56                 chckLst.Content = border;
57             }
58         }
59     }
60 }

```

```

54        };
55
56        //Create grid
57        Grid grd = new Grid
58        {
59            Name = "lst" + chckLst.Id.ToString()
60        };
61        grd.ColumnDefinitions.Add(col1);
62        grd.ColumnDefinitions.Add(col2);
63        grd.RowDefinitions.Add(new RowDefinition());
64        grd.RowDefinitions.Add(new RowDefinition());
65
66        //Create element for name
67        TextBlock textBlock = new TextBlock
68        {
69            VerticalAlignment = VerticalAlignment.Top,
70            TextWrapping = TextWrapping.Wrap,
71            Margin = new Thickness(10),
72            TextAlignment = TextAlignment.Right,
73            Text = chckLst.Name
74        };
75
76        //Create element for list
77        ListView lstView = new ListView
78        {
79            VerticalAlignment = VerticalAlignment.Top,
80            Margin = new Thickness(10),
81            Height = 26 * chckLst.ChecklistItems.Count
82        };
83        foreach (ChecklistItem item in chckLst.ChecklistItems)
84        {
85            CheckBox checkBox = new CheckBox
86            {
87                IsChecked = item.Done,
88                Content = item.NameItem,
89                Tag = item
90            };
91            checkBox.Checked += ChecklistItem_Checked;
92            lstView.Items.Add(checkBox);
93        }
94
95        grd.Children.Add(textBlock);
96        grd.Children.Add(lstView);
97        Grid.SetRowSpan(lstView, 2);
98        Grid.SetColumn(lstView, 1);
99        border.Child = grd;
100
101        lstLists.Items.Add(border);
102    }
103}
104
105 private void ChecklistItem_Checked(object sender, RoutedEventArgs e)
106 {
107     ChecklistItem item = (sender as CheckBox).Tag as ChecklistItem;
108     item.Done = (sender as CheckBox).IsChecked == true;
109     controller.UpdateCheckList(item.NameItem, item.Done, item);
110 }
111 private void BtnCancel_Click(object sender, EventArgs e)
112 {
113     Close();
114 }
115 private void BtnAddList_Click(object sender, EventArgs e)
116 {
117     ChecklistCreate checklistCreate = new ChecklistCreate();
118     if (checklistCreate.ShowDialog() == true)
119     {
120         string name = checklistCreate.tbxName.Text.Trim();
121         Checklist checklist = controller.CreateCheckList(name, userStory);
122         foreach (ChecklistItem item in checklistCreate.itemsToAdd)
123         {
124             controller.CreateCheckListItem(item.NameItem, checklist);
125         }
126         Refresh();
127     }
128 }
129
130 private void lstLists_TouchUp(object sender, EventArgs e)
131 {

```

```

133     if(lstLists.SelectedItem != null)
134     {
135         Checklist checklist = (lstLists.SelectedItem as Border).Tag as Checklist;
136         ChecklistEdit checklistEdit = new ChecklistEdit(checklist, userStory, controller);
137         if (checklistEdit.ShowDialog() == true)
138         {
139             if (checklistEdit.Deleted)
140             {
141                 controller.Delete(checklist);
142             }
143             else
144             {
145                 List<ChecklistItem> items = new List<ChecklistItem>();
146                 foreach (object item in checklistEdit.listItems.Items)
147                 {
148                     items.Add(item as ChecklistItem);
149                 }
150                 string name = checklistEdit.tbxName.Text.Trim();
151                 controller.UpdateCheckList(name, items, checklist);
152             }
153             Refresh();
154         }
155     }
156 }
157 }
158 }
159 }
```

Listing 17 – ../../Scrum'o'wall/Scrum'o'wall/Views/ChecklistMenu.xaml.cs

2.15 CommentCreate.xaml

```

1 <!--
2 * Author      :   Gaël Mariot
3 * Project     :   Scrum'o'wall
4 * File        :   CommentCreate.xaml
5 * Desc.       :   This file contains the basic template of the CommentCreate view
6 -->
7 <Window x:Class="Scrum_o_wall.Views.CommentCreate"
8     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
9     xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
10    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
11    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
12    xmlns:local="clr-namespace:Scrum_o_wall.Views"
13    mc:Ignorable="d"
14    Title="Création d'un commentaire" Height="310.865" Width="445.866" -->
15    WindowStartupLocation="CenterScreen" FontSize="16">
16     <Grid>
17         <Grid.ColumnDefinitions>
18             <ColumnDefinition Width="25*></ColumnDefinition>
19             <ColumnDefinition Width="48*></ColumnDefinition>
20         </Grid.ColumnDefinitions>
21         <Grid.RowDefinitions>
22             <RowDefinition Height="63*></RowDefinition>
23             <RowDefinition Height="117*></RowDefinition>
24             <RowDefinition Height="50*></RowDefinition>
25             <RowDefinition Height="50*></RowDefinition>
26         </Grid.RowDefinitions>
27         <TextBlock Grid.ColumnSpan="2" TextAlignment="Center" FontSize="20" -->
28             VerticalAlignment="Center" Margin="10">Poster un -->
29             commentaire</TextBlock>
30         <TextBlock Grid.Row="1" TextAlignment="Right" -->
31             VerticalAlignment="Center" Margin="10,45,10,50" Height="22" >Contenu <!--
32             :</TextBlock>
33         <TextBox x:Name="tbxContent" MaxLength="65535" Grid.Row="1" <!--
34             Grid.Column="1" TextWrapping="WrapWithOverflow" Margin="10" <!--
35             Stylus.IsPressAndHoldEnabled="False"></TextBox>
36         <TextBlock Grid.Row="2" TextAlignment="Right" <!--
37             VerticalAlignment="Center" Margin="10,12,10,18" Height="20" >Auteur <!--
38             :</TextBlock>
39         <ComboBox Name="cbxAuthor" Grid.Row="2" Grid.Column="1" Margin="10" <!--
40             Stylus.IsPressAndHoldEnabled="False">
41         </ComboBox>
42         <Button x:Name="btnCancel" Grid.Row="3" Grid.Column="0" Margin="10" <!--
43             Click="BtnCancel_Click" TouchUp="BtnCancel_Click">Annuler</Button>
```

```

33         <Button x:Name="btnConfirm" Grid.Row="3" Grid.Column="1" Margin="10" ↵
34             Click="BtnConfirm_Click" TouchUp="BtnConfirm_Click">Confirmer</Button>
35     </Grid>
36 </Window>

```

Listing 18 – ../../Scrum'o'wall/Scrum'o'wall/Views/CommentCreate.xaml

2.16 CommentCreate.xaml.cs

```

1  /*
2   * Author      : Gaël Serge Mariot
3   * Project    : Scrum'o'wall
4   * File       : CommentCreate.xaml.cs
5   * Desc.      : This file contains the logic in the CommentCreate view
6   */
7  using Scrum_o_wall.Classes;
8  using System;
9  using System.Collections.Generic;
10 using System.Windows;
11
12 namespace Scrum_o_wall.Views
13 {
14     /// <summary>
15     /// Logique d'interaction pour CommentCreate.xaml
16     /// </summary>
17     public partial class CommentCreate : Window
18     {
19         public CommentCreate(List<User> assignedUsers)
20         {
21             InitializeComponent();
22             foreach (User user in assignedUsers)
23             {
24                 cbxAuthor.Items.Add(user);
25             }
26         }
27
28         private void btnCancel_Click(object sender, EventArgs e)
29         {
30
31             Close();
32         }
33         private void BtnConfirm_Click(object sender, EventArgs e)
34         {
35             if (cbxAuthor.SelectedItem != null && tbxContent.Text.Trim().Length <=
36                 > 0)
37             {
38                 DialogResult = true;
39                 Close();
40             }
41             else
42             {
43                 MessageBox.Show("Un ou plusieurs champ(s) n'est pas rempli !", ←
44                     "Erreur", MessageBoxButtons.OK, MessageBoxIcon.Error);
45             }
46         }
47     }

```

Listing 19 – ../../Scrum'o'wall/Scrum'o'wall/Views/CommentCreate.xaml.cs

2.17 CommentMenu.xaml

```

1  <!--
2   * Author      : Gaël Mariot
3   * Project    : Scrum'o'wall
4   * File       : CommentMenu.xaml
5   * Desc.      : This file contains the basic template of the CommentMenu view
6   -->
7  <Window x:Class="Scrum_o_wall.Views.CommentMenu"
8      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
9      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
10     xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
11     xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

```

```

12     xmlns:local="clr-namespace:Scrum_o_wall.Views"
13     mc:Ignorable="d"
14     Title="Gestion des commentaires" Height="450" Width="447.5" ←
15         WindowStartupLocation="CenterScreen" FontSize="16">
16     <Grid>
17         <Grid.RowDefinitions>
18             <RowDefinition Height="70"/></RowDefinition>
19             <RowDefinition/>
20             <RowDefinition Height="50"/></RowDefinition>
21         </Grid.RowDefinitions>
22         <Grid.ColumnDefinitions>
23             <ColumnDefinition/></ColumnDefinition>
24             <ColumnDefinition/></ColumnDefinition>
25         </Grid.ColumnDefinitions>
26         <TextBlock Grid.ColumnSpan="2" VerticalAlignment="Center" ←
27             Margin="10,16" TextAlignment="Center" FontSize="28" ←
28             Height="38">Gestion des commentaires</TextBlock>
29         <Border Grid.Row="1" Grid.ColumnSpan="2" Margin="10" ←
30             BorderThickness="1" BorderBrush="{DynamicResource {x:Static ←
31                 SystemColors.GrayTextBrushKey}}">
32             <ListView x:Name="lstComments" ←
33                 MouseDoubleClick="LstComments_MouseDoubleClick" ←
34                 TouchUp="LstComments_MouseDoubleClick">
35             </ListView>
36         </Border>
37
38         <Button TouchUp="BtnCancel_Click" x:Name="btnCancel" Grid.Row="2" ←
39             Margin="10" Click="BtnCancel_Click">Retour</Button>
40         <Button TouchUp="BtnAddComment_Click" x:Name="btnAddComment" ←
41             Grid.Row="2" Grid.Column="1" Margin="10" ←
42             Click="BtnAddComment_Click">Ajouter</Button>
43     </Grid>
44 </Window>

```

Listing 20 – ../../Scrum'o'wall/Scrum'o'wall/Views/CommentMenu.xaml

2.18 CommentMenu.xaml.cs

```

1 /*
2 * Author : Gaël Serge Mariot
3 * Project : Scrum'o'wall
4 * File : CommentMenu.xaml.cs
5 * Desc. : This file contains the logic in the CommentMenu view
6 */
7 using Scrum_o_wall.Classes;
8 using System;
9 using System.Windows;
10 using System.Windows.Controls;
11
12 namespace Scrum_o_wall.Views
13 {
14     /// <summary>
15     /// Logique d'interaction pour CommentMenu.xaml
16     /// </summary>
17     public partial class CommentMenu : Window
18     {
19         private readonly UserStory userStory;
20         private readonly Controller controller;
21         public CommentMenu(UserStory aUserStory, Controller aController)
22         {
23             userStory = aUserStory;
24             controller = aController;
25
26             InitializeComponent();
27
28             Refresh();
29         }
30
31         public void Refresh()
32         {
33             lstComments.Items.Clear();
34             foreach (Comment comment in userStory.Comments)
35             {
36                 lstComments.Items.Add(comment);
37             }
38         }
39     }

```

```

40     private void BtnCancel_Click(object sender, EventArgs e)
41     {
42         Close();
43     }
44     private void BtnAddComment_Click(object sender, EventArgs e)
45     {
46         CommentCreate commentCreate = new CommentCreate(userStory.GetUsers());
47         if (userStory.GetUsers().Count == 0)
48         {
49             MessageBox.Show("Aucun utilisateur assigné", "Erreur", ←
50                             MessageBoxButtons.OK, MessageBoxIcon.Error);
51         }
52         else if (commentCreate.ShowDialog() == true)
53         {
54             string content = commentCreate.tbxContent.Text.Trim();
55             User user = commentCreate.cbxAuthor.SelectedItem as User;
56             controller.CreateComment(content, user, userStory);
57             Refresh();
58         }
59     }
60
61     private void LstComments_MouseDoubleClick(object sender, EventArgs e)
62     {
63         ListBox lbx = sender as ListBox;
64         if (lbx.SelectedItem != null)
65         {
66             Comment comment = lbx.SelectedItem as Comment;
67             MessageBox.Show(string.Format("Auteur : {0}\nDate : {1}\n{2}", ←
68                             comment.User, comment.DateTime, comment.Description), ←
69                             "Contenu du commentaire", MessageBoxButtons.OK);
70         }
71     }

```

Listing 21 – ../../Scrum'o'wall/Scrum'o'wall/Views/CommentMenu.xaml.cs

2.19 FileCreate.xaml

```

1 <!--
2 * Author    :   Gael Mariot
3 * Project   :   Scrum'o'wall
4 * File      :   FileCreate.xaml
5 * Desc.     :   This file contains the basic template of the FileCreate view
6 -->
7 <Window x:Class="Scrum_o_wall.Views.FileCreate"
8     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
9     xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
10    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
11    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
12    xmlns:local="clr-namespace:Scrum_o_wall.Views"
13    mc:Ignorable="d"
14    Title="Ajouter un fichier" Height="500" Width="600" ←
15        WindowStartupLocation="CenterScreen" FontSize="16">
16     <Grid>
17         <Grid.RowDefinitions>
18             <RowDefinition Height="78*></RowDefinition>
19             <RowDefinition Height="56*></RowDefinition>
20             <RowDefinition Height="227*></RowDefinition>
21             <RowDefinition Height="56*></RowDefinition>
22         </Grid.RowDefinitions>
23         <Grid.ColumnDefinitions>
24             <ColumnDefinition Width="140"></ColumnDefinition>
25             <ColumnDefinition/>
26         </Grid.ColumnDefinitions>
27         <TextBlock Grid.ColumnSpan="2" VerticalAlignment="Center" ←
28             Margin="10,26" TextAlignment="Center" FontSize="20" Height="26" ←
29             >Ajouter un fichier</TextBlock>
30         <TextBlock Grid.Row="1" VerticalAlignment="Center" Margin="10,14,10,22" ←
31             TextAlignment="Right" Height="20" ><Run Text="Fichier à ajouter :"/><LineBreak/><Run/><LineBreak/><Run/></TextBlock>
32         <TextBlock Grid.Row="2" VerticalAlignment="Top" Margin="10,10,10,0" ←
33             TextAlignment="Right" Height="21" ><Description :/></TextBlock>
34         <TextBox x:Name="txbFileName" MaxLength="65535" IsEnabled="False" ←
35             Grid.Row="1" Grid.Column="1" Margin="10,10,100,10" ></TextBox>

```

```

30         <Button x:Name="btnFileSearch" TouchUp="BtnFileSearch_Click" ↵
31             Grid.Row="1" Grid.Column="1" Margin="0,10,10,10" ↵
32             Click="BtnFileSearch_Click" HorizontalAlignment="Right" ↵
33             Width="87">+</Button>
34         <TextBox Margin="10" MaxLength="65535" TextWrapping="Wrap" ↵
35             Name="tbxDescription" Grid.Row="2" Grid.Column="1" ↵
36             Stylus.IsPressAndHoldEnabled="False">
37     </TextBox>
38
39         <Button TouchUp="BtnCancel_Click" x:Name="btnCancel" Grid.Row="4" ↵
40             Margin="10" Click="BtnCancel_Click">Annuler</Button>
41         <Button TouchUp="BtnConfirm_Click" x:Name="btnConfirm" Grid.Row="4" ↵
42             Grid.Column="1" Margin="10" Click="BtnConfirm_Click">Confirmer</Button>
43     </Grid>
44 </Window>

```

Listing 22 – .../Scrum'o'wall/Scrum'o'wall/Views/FileCreate.xaml

2.20 FileCreate.xaml.cs

```

1 /*
2  * Author      : Gaël Serge Mariot
3  * Project     : Scrum'o'wall
4  * File        : FileCreate.xaml.cs
5  * Desc.       : This file contains the logic in the FileCreate view
6 */
7 using Microsoft.Win32;
8 using System;
9 using System.Windows;
10
11 namespace Scrum_o_wall.Views
12 {
13     /// <summary>
14     /// Logique d'interaction pour FileCreate.xaml
15     /// </summary>
16     public partial class FileCreate : Window
17     {
18         public FileCreate()
19         {
20             InitializeComponent();
21         }
22
23         private void BtnFileSearch_Click(object sender, EventArgs e)
24         {
25             OpenFileDialog opf = new OpenFileDialog
26             {
27                 Multiselect = false
28             };
29             if (opf.ShowDialog() == true)
30             {
31                 tbxFileName.Text = opf.FileName;
32             }
33         }
34         private void btnCancel_Click(object sender, EventArgs e)
35         {
36             Close();
37         }
38         private void btnConfirm_Click(object sender, EventArgs e)
39         {
40             int descriptionLength = tbxDescription.Text.Trim().Length;
41             int fileNameLength = tbxFileName.Text.Trim().Length;
42             if (descriptionLength > 0 && fileNameLength > 0 && ←
43                 System.IO.File.Exists(tbxFileName.Text))
44             {
45                 DialogResult = true;
46                 Close();
47             }
48             else
49             {
50                 MessageBox.Show("Un ou plusieurs champ(s) n'est pas rempli!", ←
51                             "Erreur", MessageBoxButtons.OK, MessageBoxIcon.Error);
52             }
53         }
54     }

```

Listing 23 – ../../Scrum'o'wall/Scrum'o'wall/Views/FileCreate.xaml.cs

2.21 FileEdit.xaml

```
1 <!--
2 * Author    :   Ga l Mariot
3 * Project   :   Scrum'o'wall
4 * File      :   FileEdit.xaml
5 * Desc.     :   This file contains the basic template of the FileEdit view
6 -->
7 <Window x:Class="Scrum_o_wall.Views.FileEdit"
8     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
9     xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
10    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
11    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
12    xmlns:local="clr-namespace:Scrum_o_wall.Views"
13    mc:Ignorable="d"
14    Title="Modifier un fichier" Height="500" Width="600" ↵
15        WindowStartupLocation="CenterScreen" FontSize="16">
16    <Grid>
17        <Grid.RowDefinitions>
18            <RowDefinition Height="70*></RowDefinition>
19            <RowDefinition Height="50*></RowDefinition>
20            <RowDefinition Height="208*></RowDefinition>
21            <RowDefinition Height="50*></RowDefinition>
22        </Grid.RowDefinitions>
23        <Grid.ColumnDefinitions>
24            <ColumnDefinition Width="140"></ColumnDefinition>
25            <ColumnDefinition/>
26        </Grid.ColumnDefinitions>
27        <TextBlock Grid.ColumnSpan="2" VerticalAlignment="Center" Margin="10" ↵
28            TextAlignment="Center" FontSize="20" >Modifier un fichier</TextBlock>
29        <TextBlock Grid.Row="1" VerticalAlignment="Center" Margin="10" ↵
30            TextAlignment="Right" >Fichier :</TextBlock>
31        <TextBlock Grid.Row="2" VerticalAlignment="Top" Margin="10" ↵
32            TextAlignment="Right" >Description :</TextBlock>
33        <TextBox x:Name="txbFileName" MaxLength="65535" IsEnabled="False" ↵
34            Grid.Row="1" Grid.Column="1" Margin="10" ></TextBox>
35        <TextBox Margin="10" MaxLength="65535" TextWrapping="Wrap" ↵
36            Name="txbDescription" Grid.Row="2" Grid.Column="1" ↵
37            Stylus.IsPressAndHoldEnabled="False"/>
38
39        <Grid Grid.Row="4" Grid.ColumnSpan="3">
40            <Grid.ColumnDefinitions>
41                <ColumnDefinition Width="1*></ColumnDefinition>
42                <ColumnDefinition Width="1*></ColumnDefinition>
43                <ColumnDefinition Width="3*></ColumnDefinition>
44            </Grid.ColumnDefinitions>
45            <Button TouchUp="BtnCancel_Click" x:Name="btnCancel" Margin="10" ↵
46                Click="BtnCancel_Click" >Annuler</Button>
47            <Button TouchUp="BtnDelete_Click" x:Name="btnDelete" ↵
48                Grid.Column="1" Margin="10" >
49                Click="BtnDelete_Click" >Supprimer</Button>
50            <Button TouchUp="BtnConfirm_Click" x:Name="btnConfirm" ↵
51                Grid.Column="2" Margin="10" >
52                Click="BtnConfirm_Click" >Confirmer</Button>
53        </Grid>
54    </Grid>
55 </Window>
```

Listing 24 – ../../Scrum'o'wall/Scrum'o'wall/Views/FileEdit.xaml

2.22 FileEdit.xaml.cs

```
1 /*
2 * Author    :   Ga l Serge Mariot
3 * Project   :   Scrum'o'wall
4 * File      :   FileEdit.xaml.cs
5 * Desc.     :   This file contains the logic in the FileEdit view
6 */
7 using Scrum_o_wall.Classes;
```

```

8  using System;
9  using System.Windows;
10
11 namespace Scrum_o_wall.Views
12 {
13     /// <summary>
14     /// Logique d'interaction pour FileEdit.xaml
15     /// </summary>
16     public partial class FileEdit : Window
17     {
18         private readonly File file;
19         public bool Deleted = false;
20
21         public FileEdit(File aFile)
22         {
23             file = aFile;
24
25             InitializeComponent();
26
27             tbxDescription.Text = file.Description;
28             tbxFileName.Text = file.Name;
29         }
30
31         private void btnCancel_Click(object sender, EventArgs e)
32         {
33
34             Close();
35         }
36         private void btnConfirm_Click(object sender, EventArgs e)
37         {
38
39             int descriptionLength = tbxDescription.Text.Trim().Length;
40             if (descriptionLength > 0)
41             {
42                 DialogResult = true;
43                 Close();
44             }
45             else
46             {
47                 MessageBox.Show("Un ou plusieurs champ(s) n'est pas rempli !", ←
48                             "Erreur", MessageBoxButtons.OK, MessageBoxIcon.Error);
49             }
50
51         private void.btnDelete_Click(object sender, EventArgs e)
52         {
53             if (MessageBox.Show("Le fichier sera supprim .\n veux-tu vraiment?", "Attention", MessageBoxButtons.YesNo, MessageBoxIcon.Warning) == DialogResult.Yes)
54             {
55                 DialogResult = true;
56                 Deleted = true;
57                 Close();
58             }
59         }
60     }
61 }
```

Listing 25 – ../../Scrum'o'wall/Scrum'o'wall/Views/FileEdit.xaml.cs

2.23 FileMenu.xaml

```

1  <!--
2  * Author    :   Ga l Mariot
3  * Project   :   Scrum'o'wall
4  * File      :   FileMenu.xaml
5  * Desc.     :   This file contains the basic template of the FileMenu view
6  -->
7 <Window x:Class="Scrum_o_wall.Views.FileMenu"
8     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
9     xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
10    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
11    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
12    xmlns:local="clr-namespace:Scrum_o_wall.Views"
13    mc:Ignorable="d"
14    Title="Gestion des fichiers" Height="450" Width="447.5" ←
15        WindowStartupLocation="CenterScreen" FontSize="16">
```

```

15    <Grid>
16        <Grid.RowDefinitions>
17            <RowDefinition Height="70"/></RowDefinition>
18            <RowDefinition/>
19            <RowDefinition Height="50"/></RowDefinition>
20    </Grid.RowDefinitions>
21    <Grid.ColumnDefinitions>
22        <ColumnDefinition/></ColumnDefinition>
23        <ColumnDefinition/></ColumnDefinition>
24    </Grid.ColumnDefinitions>
25    <TextBlock Grid.ColumnSpan="2" VerticalAlignment="Center" ←
26        Margin="10,16" TextAlignment="Center" FontSize="28" ←
27        Height="38">Gestion des fichiers</TextBlock>
28    <Border Grid.Row="1" Grid.ColumnSpan="2" Margin="10" ←
29        BorderThickness="1" BorderBrush="{DynamicResource{x:Static←
30        SystemColors.GrayTextBrushKey}}">
31        <ListView x:Name="lstFiles" TouchUp="LstFiles_MouseDoubleClick" ←
32            MouseDoubleClick="LstFiles_MouseDoubleClick" ←
33            Stylus.IsPressAndHoldEnabled="False">
34        </ListView>
35    </Border>
36
37    <Button TouchUp="Quit_Click" x:Name="btnCancel" Grid.Row="2" ←
38        Margin="10" Click="Quit_Click">Retour</Button>
39    <Button TouchUp="BtnAddFile_Click" x:Name="btnAddFile" Grid.Row="2" ←
40        Grid.Column="1" Margin="10" Click="BtnAddFile_Click">Ajouter</Button>
41
42 </Grid>
43 </Window>

```

Listing 26 – ../../Scrum'o'wall/Scrum'o'wall/Views/FileMenu.xaml

2.24 FileMenu.xaml.cs

```

1 /*
2 * Author : Ga l Serge Mariot
3 * Project : Scrum'o'wall
4 * File : FileMenu.xaml.cs
5 * Desc. : This file contains the logic in the FileMenu view
6 */
7 using Scrum_o_wall.Classes;
8 using System;
9 using System.Windows;
10 using System.Windows.Controls;
11 using System.Windows.Media;
12
13 namespace Scrum_o_wall.Views
14 {
15     /// <summary>
16     /// Logique d'interaction pour FileMenu.xaml
17     /// </summary>
18     public partial class FileMenu : Window
19     {
20         private readonly UserStory userStory;
21         private readonly Controller controller;
22         public FileMenu(UserStory aUserStory, Controller aController)
23         {
24             userStory = aUserStory;
25             controller = aController;
26
27             InitializeComponent();
28
29             Refresh();
30         }
31
32         public void Refresh()
33         {
34             lstFiles.Items.Clear();
35             foreach (File file in userStory.Files)
36             {
37                 //Create border
38                 Border border = new Border
39                 {
40                     BorderBrush = Brushes.Black,
41                     BorderThickness = new Thickness(1),
42                     Width = 408,
43                     Tag = file
44                 }
45             }
46         }
47     }
48 }

```

```

45    };
46    border.TouchDown += FileInList_Click;
47    border.MouseLeftButtonDown += FileInList_Click;
48
49    //Create grid
50    Grid grd = new Grid
51    {
52        Name = "lst" + file.Id.ToString()
53    };
54    grd.RowDefinitions.Add(new RowDefinition());
55    grd.RowDefinitions.Add(new RowDefinition());
56
57    //Create element for name
58    TextBlock textBlock = new TextBlock
59    {
60        VerticalAlignment = VerticalAlignment.Top,
61        TextWrapping = TextWrapping.Wrap,
62        Margin = new Thickness(10),
63        TextAlignment = TextAlignment.Right,
64        Text = file.Name
65    };
66
67    //Create element for name
68    TextBlock descBlock = new TextBlock
69    {
70        VerticalAlignment = VerticalAlignment.Top,
71        TextWrapping = TextWrapping.Wrap,
72        Margin = new Thickness(10),
73        TextAlignment = TextAlignment.Right,
74        Text = file.Name
75    };
76
77    grd.Children.Add(textBlock);
78    grd.Children.Add(descBlock);
79    Grid.SetColumn(descBlock, 1);
80    border.Child = grd;
81
82    lstFiles.Items.Add(file);
83}
84
85
86 private void FileInList_Click(object sender, EventArgs e)
87{
88    File file = (sender as Border).Tag as File;
89    FileEdit fileEdit = new FileEdit(file);
90    if (fileEdit.ShowDialog() == true)
91    {
92        if (fileEdit.Deleted)
93        {
94            controller.Delete(file);
95        }
96        else
97        {
98            controller.UpdateFile(fileEdit.tbxDescription.Text.Trim(), ←
99                           file);
100        }
101        Refresh();
102    }
103    if (fileEdit.ShowDialog() == true)
104    {
105        controller.UpdateFile(fileEdit.tbxDescription.Text.Trim(), file);
106    }
107    Refresh();
108}
109
110 private void Quit_Click(object sender, EventArgs e)
111{
112    Close();
113}
114
115 private void BtnAddFile_Click(object sender, EventArgs e)
116{
117    FileCreate fileCreate = new FileCreate();
118    if (fileCreate.ShowDialog() == true)
119    {
120        string fileName = fileCreate.tbxFileName.Text.Trim();
121        string description = fileCreate.tbxDescription.Text.Trim();
122        controller.CreateFile(fileName, description, userStory);
123        Refresh();
124    }

```

```

123
124 }
125
126     private void LstFiles_MouseDoubleClick(object sender, EventArgs e)
127     {
128         ListBox lbx = sender as ListBox;
129         if (lbx.SelectedItem != null)
130         {
131             File file = lbx.SelectedItem as File;
132             FileEdit fileEdit = new FileEdit(file);
133             if (fileEdit.ShowDialog() == true)
134             {
135                 if (fileEdit.Deleted)
136                 {
137                     controller.Delete(file);
138                 }
139                 else
140                 {
141                     controller.UpdateFile(fileEdit.tbxDescription.Text, file);
142                 }
143             }
144         }
145     }
146 }
147
148 }
```

Listing 27 – ../../Scrum'o'wall/Scrum'o'wall/Views/FileMenu.xaml.cs

2.25 MainMenu.xaml

```

1 <!--
2 * Author      :   Ga l Mariot
3 * Project     :   Scrum'o'wall
4 * File        :   ActivitiesMenu.xaml
5 * Desc.       :   This file contains the basic template of the ActivitiesMenu view
6 -->
7 <Window x:Class="Scrum_o_wall.MainMenu"
8     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
9     xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
10    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
11    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
12    xmlns:local="clr-namespace:Scrum_o_wall"
13    mc:Ignorable="d"
14    Title="Scrum'o'Wall" ResizeMode="NoResize" WindowState="Maximized" ←
15        WindowStartupLocation="CenterScreen" Width="1000" Height="1000" ←
16        WindowStyle="None" FontSize="16">
17     <Window.ContextMenu >
18         <ContextMenu >
19             <MenuItem Header="Ajouter un projet" Name="AddProject" ←
20                 Click="AddProject_Click"/>
21             <Separator/>
22             <MenuItem Header="Quitter" Name="Quit" Click="Quit_Click"/>
23     </Window.ContextMenu>
24     <Grid>
25         <Grid.RowDefinitions >
26             <RowDefinition />
27             <RowDefinition Height="100" />
28         </Grid.RowDefinitions>
29         <Grid.ColumnDefinitions>
30             <ColumnDefinition></ColumnDefinition>
31             <ColumnDefinition></ColumnDefinition>
32         </Grid.ColumnDefinitions>
33         <ScrollViewer x:Name="scrllVwr" VerticalScrollBarVisibility="Auto" ←
34             Grid.ColumnSpan="2">
35             <Canvas x:Name="cnvsProject" Width="1000" Margin="0,0,0,0" ←
36                 HorizontalAlignment="Left" Height="796">
37                 </Canvas>
38         </ScrollViewer>
39         <Button Grid.Row="1" Grid.Column="1" HorizontalAlignment="Left" ←
40             x:Name="btnAddProject" TouchUp="AddProject_Click" ←
41             Click="AddProject_Click" Content="+" VerticalContentAlignment="Top" ←
42             Width="75" Height="75" BorderBrush="Black" FontSize="45" Margin="10">
43             <Button.Resources>
44                 <Style TargetType="Border">
45                     <Setter Property="CornerRadius" Value="50"/>
```

```

39         </Style>
40     </Button.Resources>
41   </Button>
42   <Button Grid.Row="1" HorizontalAlignment="Right" x:Name="btnReturn" ↵
43     TouchUp="Quit_Click" Click="Quit_Click" Content="Ã " ↵
44     VerticalContentAlignment="Top" BorderBrush="Black" FontSize="45" ↵
45     Width="75" Height="75" Margin="10">
46     <Button.Resources>
47       <Style TargetType="{x:Type Border}">
48         <Setter Property="CornerRadius" Value="50"/>
49       </Style>
50     </Button.Resources>
51   </Button>
52 </Grid>
53 </Window>

```

Listing 28 – ../../Scrum'o'wall/Scrum'o'wall/Views/MainMenu.xaml

2.26 MainMenu.xaml.cs

```

1 /*
2 * Author : Gaël Serge Mariot
3 * Project : Scrum'o'wall
4 * File : MainMenu.xaml.cs
5 * Desc. : This file contains the logic in the MainMenu view
6 */
7 using Scrum_o_wall.Classes;
8 using Scrum_o_wall.Views;
9 using System;
10 using System.Collections.Generic;
11 using System.Windows;
12 using System.Windows.Controls;
13 using System.Windows.Input;
14 using System.Windows.Media;
15
16 namespace Scrum_o_wall
17 {
18     /// <summary>
19     /// Logique d'interaction pour MainMenu.xaml
20     /// </summary>
21     public partial class MainMenu : Window
22     {
23         private readonly Controller controller;
24         private readonly List<UserControl> projectControls = new ←
25             List<UserControl>();
26         public MainMenu()
27         {
28             InitializeComponent();
29             controller = new Controller();
30             Loaded += MainMenu_Loaded;
31         }
32
33         private void Refresh()
34         {
35             //Create controls for the projects
36             int maxProj = controller.Projects.Count;
37             foreach (UserControl project in projectControls)
38             {
39                 cnvsProject.Children.Remove(project);
40             }
41             projectControls.Clear();
42             for (int i = 0; i < maxProj; i++)
43             {
44                 Project p = controller.Projects[i];
45
46                 //create a control for title
47                 Label title = new Label();
48                 {
49                     Content = p.Name,
50                     HorizontalContentAlignment = HorizontalAlignment.Center,
51                     FontSize = 24
52                 };
53                 Grid.SetRow(title, 0);
54
55                 //Create a control for description
56                 Label desc = new Label();
57                 TextBlock txtBlck = new TextBlock

```

```

57
58     {
59         TextWrapping = TextWrapping.WrapWithOverflow,
60         Text = p.Description
61     };
62     desc.Content = txtBlck;
63     desc.HorizontalAlignment = HorizontalAlignment.Stretch;
64     desc.FontSize = 18;
65     Grid.SetRow(desc, 1);
66
67     //Create a control of date
68     Label date = new Label
69     {
70         Content = "Date added but : " + p.Begin.ToShortDateString(),
71         HorizontalContentAlignment = HorizontalAlignment.Left,
72         FontSize = 18
73     };
74     Grid.SetRow(date, 2);
75
76     //Place controls in a grid
77     Grid grd = new Grid();
78     grd.RowDefinitions.Add(new RowDefinition() { Height = new GridLength(60) });
79     grd.RowDefinitions.Add(new RowDefinition());
80     grd.RowDefinitions.Add(new RowDefinition() { Height = new GridLength(40) });
81     grd.Children.Add(title);
82     grd.Children.Add(desc);
83     grd.Children.Add(date);
84
85     //Create project frame
86     UserControl usrCntrl = new UserControl
87     {
88         Width = cnvsProject.Width / 4,
89         Height = cnvsProject.Height / 5,
90         Content = grd,
91         BorderBrush = Brushes.Black,
92         Background = Brushes.LightGray,
93         Cursor = Cursors.Hand,
94         Tag = p
95     };
96     usrCntrl.TouchDown += UsrCntrl_Click;
97     usrCntrl.MouseLeftButtonDown += UsrCntrl_Click;
98
99     //Positioning of control
100    Canvas.SetLeft(usrCntrl, (cnvsProject.Width - usrCntrl.Width) / 2.0 + ((usrCntrl.Width + usrCntrl.Width / 4) * (i % 3 - 1)));
101    Canvas.SetTop(usrCntrl, 20 + (usrCntrl.Height + usrCntrl.Height / 5) * (i / 3));
102
103    if (Canvas.GetTop(usrCntrl) + usrCntrl.Height > cnvsProject.Height)
104    {
105        cnvsProject.Height = Canvas.GetTop(usrCntrl) + usrCntrl.Height;
106    }
107
108    //Add project frame to canvas
109    cnvsProject.Children.Add(usrCntrl);
110    projectControls.Add(usrCntrl);
111 }
112
113 private void MainMenu_Loaded(object sender, RoutedEventArgs e)
114 {
115     //ActualWidth and ActualHeight measured when window is loaded minus border sizes
116     cnvsProject.Width = ActualWidth;
117     cnvsProject.Height = ActualHeight;
118     scrllVwr.Width = cnvsProject.Width;
119     scrllVwr.Height = cnvsProject.Height;
120
121     //Refresh the view
122     Refresh();
123 }
124 private void UsrCntrl_Click(object sender, EventArgs e)
125 {
126     Project p = (sender as UserControl).Tag as Project;
127     ProjectMenu projectMenu = new ProjectMenu(p, controller);
128     projectMenu.ShowDialog();

```

```

129         Refresh();
130     }
131     private void Quit_Click(object sender, EventArgs e)
132     {
133         if (MessageBox.Show("Vous allez quitter l'application.\n tes -vous ↵
134             s r de vouloir continuer?", "Attention", ↵
135             MessageBoxButtons.OKCancel, MessageBoxIcon.Warning) == ↵
136             MessageBoxResult.OK)
137         {
138             Application.Current.Shutdown();
139         }
140     }
141     private void AddProject_Click(object sender, EventArgs e)
142     {
143         ProjectCreate projectCreate = new ProjectCreate();
144         if (projectCreate.ShowDialog() == true)
145         {
146             string name = projectCreate.tbxName.Text;
147             string desc = projectCreate.tbxDesc.Text;
148             DateTime date = (DateTime)projectCreate.tbxDate.SelectedDate;
149             controller.CreateProject(name, desc, date);
150
151             Refresh();
152         }
153     }

```

Listing 29 – ../../Scrum'o'wall/Scrum'o'wall/Views/MainMenu.xaml.cs

2.27 MindmapCreate.xaml

```

1 <!--
2 * Author : Ga l Mariot
3 * Project : Scrum'o'wall
4 * File : MindmapCreate.xaml
5 * Desc. : This file contains the basic template of the MindmapCreate view
6 -->
7 <Window x:Class="Scrum_o_wall.Views.MindmapCreate"
8     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
9     xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
10    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
11    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
12    xmlns:local="clr-namespace:Scrum_o_wall.Views"
13    mc:Ignorable="d"
14    Title="Cr ation d'un mindmap" Height="196.027" Width="538.14" ↵
15        WindowStartupLocation="CenterScreen" FontSize="16">
16     <Grid>
17         <Grid.ColumnDefinitions>
18             <ColumnDefinition Width="161" /></ColumnDefinition>
19             <ColumnDefinition /></ColumnDefinition>
20         </Grid.ColumnDefinitions>
21         <Grid.RowDefinitions>
22             <RowDefinition Height="47*" /></RowDefinition>
23             <RowDefinition Height="50*" /></RowDefinition>
24             <RowDefinition Height="62*" /></RowDefinition>
25         </Grid.RowDefinitions>
26         <TextBlock Grid.ColumnSpan="2" TextAlignment="Center" FontSize="20" ↵
27             VerticalAlignment="Center" Height="28" Margin="0,10,0,9">Cr ation ↵
28             d'un mindmap</TextBlock>
29         <TextBlock Grid.Row="1" TextAlignment="Right" ↵
30             VerticalAlignment="Center" Margin="10">Nom :</TextBlock>
31         <TextBox x:Name="tbxName" Grid.Row="1" MaxLength="100" Grid.Column="1" ↵
32             Margin="10" Stylus.IsPressAndHoldEnabled="False" /></TextBox>
33         <Button TouchUp="BtnCancel_Click" x:Name="btnCancel" Grid.Row="2" ↵
34             Grid.Column="0" Margin="10" Click="BtnCancel_Click">Annuler</Button>
35         <Button TouchUp="BtnConfirm_Click" x:Name="btnConfirm" Grid.Row="2" ↵
36             Grid.Column="1" Margin="10" Click="BtnConfirm_Click">Confirmer</Button>
37     </Grid>
38 </Window>

```

Listing 30 – ../../Scrum'o'wall/Scrum'o'wall/Views/MindmapCreate.xaml

2.28 MindmapCreate.xaml.cs

```

1  /*
2   * Author    : Gaël Serge Mariot
3   * Project   : Scrum'o'wall
4   * File      : MindmapCreate.xaml.cs
5   * Desc.     : This file contains the logic in the MindmapCreate view
6   */
7  using System;
8  using System.Windows;
9
10 namespace Scrum_o_wall.Views
11 {
12     /// <summary>
13     /// Logique d'interaction pour MindmapCreate.xaml
14     /// </summary>
15     public partial class MindmapCreate : Window
16     {
17         public MindmapCreate()
18         {
19             InitializeComponent();
20         }
21
22         private void btnCancel_Click(object sender, EventArgs e)
23         {
24
25             Close();
26         }
27         private void btnConfirm_Click(object sender, EventArgs e)
28         {
29             if (tbxName.Text.Trim().Length > 0)
30             {
31                 DialogResult = true;
32                 Close();
33             }
34             else
35             {
36                 MessageBox.Show("Un ou plusieurs champ(s) n'est pas rempli !", ←
37                             "Erreur", MessageBoxButtons.OK, MessageBoxIcon.Error);
38             }
39         }
40     }

```

Listing 31 – ../../Scrum'o'wall/Scrum'o'wall/Views/MindmapCreate.xaml.cs

2.29 MindmapEdit.xaml

```

1  <!--
2   * Author    : Gaël Mariot
3   * Project   : Scrum'o'wall
4   * File      : MindmapEdit.xaml
5   * Desc.     : This file contains the basic template of the MindmapEdit view
6   -->
7  <Window x:Class="Scrum_o_wall.Views.MindmapEdit"
8      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
9      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
10     xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
11     xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
12     xmlns:local="clr-namespace:Scrum_o_wall.Views"
13     mc:Ignorable="d"
14     Title="Modification d'un mindmap" Height="196.027" Width="538.14" ←
15           WindowStartupLocation="CenterScreen" FontSize="16">
16     <Grid>
17       <Grid.ColumnDefinitions>
18         <ColumnDefinition Width="161"></ColumnDefinition>
19         <ColumnDefinition></ColumnDefinition>
20       </Grid.ColumnDefinitions>
21       <Grid.RowDefinitions>
22         <RowDefinition Height="47*></RowDefinition>
23         <RowDefinition Height="50*></RowDefinition>
24         <RowDefinition Height="62*></RowDefinition>
25       </Grid.RowDefinitions>
26       <TextBlock Grid.ColumnSpan="2" TextAlignment="Center" FontSize="20" ←
27           VerticalAlignment="Center" Height="28" ←
28           Margin="0,10,0,9">Modification d'un mindmap</TextBlock>
29       <TextBlock Grid.Row="1" TextAlignment="Right" ←
30           VerticalAlignment="Center" Margin="10">Nom :</TextBlock>

```

```

27      <TextBox x:Name="tbxName" Grid.Row="1" MaxLength="100" Grid.Column="1" ←
28          Margin="10" Stylus.IsPressAndHoldEnabled="False"></TextBox>
29      <Grid Grid.Row="2" Grid.ColumnSpan="2">
30          <Grid.ColumnDefinitions>
31              <ColumnDefinition></ColumnDefinition>
32              <ColumnDefinition></ColumnDefinition>
33              <ColumnDefinition Width="2*"></ColumnDefinition>
34          </Grid.ColumnDefinitions>
35          <Button TouchUp="BtnCancel_Click" x:Name="btnCancel" ←
36              Grid.Column="0" Margin="10" ←
37              Click="BtnCancel_Click">Annuler</Button>
38          <Button TouchUp="BtnDelete_Click" x:Name="btnDelete" ←
39              Grid.Column="1" Margin="10" ←
40              Click="BtnDelete_Click">Supprimer</Button>
41          <Button TouchUp="BtnConfirm_Click" x:Name="btnConfirm" ←
42              Grid.Column="2" Margin="10" ←
43              Click="BtnConfirm_Click">Confirmer</Button>
44      </Grid>
45  </Grid>
46 </Window>

```

Listing 32 – ../../Scrum'o'wall/Scrum'o'wall/Views/MindmapEdit.xaml

2.30 MindmapEdit.xaml.cs

```

1  /*
2  * Author     :    Gael Serge Mariot
3  * Project    :    Scrum'o'wall
4  * File       :    MindmapEdit.xaml.cs
5  * Desc.      :    This file contains the logic in the MindmapEdit view
6  */
7  using Scrum_o_wall.Classes;
8  using System;
9  using System.Windows;
10
11 namespace Scrum_o_wall.Views
12 {
13     /// <summary>
14     /// Logique d'interaction pour MindmapCreate.xaml
15     /// </summary>
16     public partial class MindmapEdit : Window
17     {
18         public bool Deleted = false;
19         private MindMap mindMap;
20         public MindmapEdit(MindMap aMindMap)
21         {
22             mindMap = aMindMap;
23             InitializeComponent();
24             tbxName.Text = mindMap.Name;
25         }
26
27         private void BtnCancel_Click(object sender, EventArgs e)
28         {
29
30             Close();
31         }
32         private void BtnDelete_Click(object sender, EventArgs e)
33         {
34             if (MessageBox.Show("Le mindmap sera supprimé.\nVoulez-vous le faire?", "Attention", MessageBoxButton.YesNo, MessageBoxImage.Warning) == MessageBoxResult.Yes)
35             {
36                 Deleted = true;
37                 DialogResult = true;
38                 Close();
39             }
40         }
41         private void BtnConfirm_Click(object sender, EventArgs e)
42         {
43             if (tbxName.Text.Trim().Length > 0)
44             {
45                 DialogResult = true;
46                 Close();
47             }
48             else
49             {

```

```

50             MessageBox.Show("Un ou plusieurs champ(s) n'est pas rempli!", ←
51                         "Erreur", MessageBoxButtons.OK, MessageBoxIcon.Error);
52         }
53     }
54 }
```

Listing 33 – ../../Scrum'o'wall/Scrum'o'wall/Views/MindmapEdit.xaml.cs

2.31 MindmapMenu.xaml

```

1 <!--
2 * Author    :   Gael Mariot
3 * Project   :   Scrum'o'wall
4 * File      :   MindmapMenu.xaml
5 * Desc.     :   This file contains the basic template of the MindmapMenu view
6 -->
7 <Window x:Class="Scrum_o_wall.Views.MindmapMenu"
8     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
9     xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
10    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
11    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
12    xmlns:local="clr-namespace:Scrum_o_wall.Views"
13    mc:Ignorable="d"
14    Title="Scrum'o'Wall" Height="450" Width="800" WindowState="Maximized" ←
15        WindowStyle="None" WindowStartupLocation="CenterScreen" ←
16        ResizeMode="NoResize" FontSize="16">
17     <Window.ContextMenu>
18         <ContextMenu>
19             <MenuItem Header="Créer un noeud" Name="CreateNode" ←
20                 Click="CreateNode_Click"/>
21             <MenuItem Header="Modifier le mindmap" Name="Modify" ←
22                 Click="Modify_Click"/>
23             <Separator/>
24             <MenuItem Header="Retour" Name="Quit" Click="Quit_Click"/>
25         </ContextMenu>
26     </Window.ContextMenu>
27     <Grid>
28         <Grid.RowDefinitions>
29             <RowDefinition />
30             <RowDefinition Height="100"/>
31         </Grid.RowDefinitions>
32         <Grid.ColumnDefinitions>
33             <ColumnDefinition/></ColumnDefinition>
34             <ColumnDefinition/></ColumnDefinition>
35         </Grid.ColumnDefinitions>
36         <ScrollViewer x:Name="scrlVwr" VerticalScrollBarVisibility="Auto" ←
37             HorizontalScrollBarVisibility="Auto" Grid.ColumnSpan="2">
38             <Grid x:Name="grdMindMap" >
39             </Grid>
40         </ScrollViewer>
41         <Button Grid.Row="1" Grid.Column="1" HorizontalAlignment="Left" ←
42             x:Name="btnAddNode" TouchUp="CreateNode_Click" ←
43             Click="CreateNode_Click" Content="+" VerticalContentAlignment="Top" ←
44             Width="75" Height="75" BorderBrush="Black" FontSize="45" Margin="10">
45             <Button.Resources>
46                 <Style TargetType="Border">
47                     <Setter Property="CornerRadius" Value="50"/>
48                 </Style>
49             </Button.Resources>
50         <Button Grid.Row="1" HorizontalAlignment="Right" x:Name="btnReturn" ←
51             TouchUp="Quit_Click" Click="Quit_Click" Content="à" ←
52             VerticalContentAlignment="Top" BorderBrush="Black" FontSize="45" ←
53             Width="75" Height="75" Margin="10">
54             <Button.Resources>
55                 <Style TargetType="{x:Type Border}">
56                     <Setter Property="CornerRadius" Value="50"/>
57                 </Style>
58             </Button.Resources>
59         </Button>
60     </Grid>
61 </Window>
```

Listing 34 – ../../Scrum'o'wall/Scrum'o'wall/Views/MindmapMenu.xaml

2.32 MindmapMenu.xaml.cs

```
1  /*
2   * Author    :    Ga l Serge Mariot
3   * Project   :    Scrum'o'wall
4   * File      :    MindmapMenu.xaml.cs
5   * Desc.     :    This file contains the logic in the MindmapMenu view
6   */
7  using Scrum_o_wall.Classes;
8  using System;
9  using System.Collections.Generic;
10 using System.Windows;
11 using System.Windows.Controls;
12 using System.Windows.Input;
13 using System.Windows.Media;
14
15 namespace Scrum_o_wall.Views
16 {
17     /// <summary>
18     /// Logique d'interaction pour MindmapMenu.xaml
19     /// </summary>
20     public partial class MindmapMenu : Window
21     {
22         private readonly MindMap mindMap;
23         private readonly Controller controller;
24         private readonly List<UserControl> nodeControls = new List<UserControl>();
25         public MindmapMenu(MindMap aMindMap, Controller aController)
26         {
27             mindMap = aMindMap;
28             controller = aController;
29
30             InitializeComponent();
31
32             Loaded += MindmapMenu_Loaded;
33
34         }
35         private void MindmapMenu_Loaded(object sender, RoutedEventArgs e)
36         {
37             Refresh();
38         }
39
40         public void Refresh()
41         {
42             foreach (UserControl nodeControl in nodeControls)
43             {
44                 grdMindMap.Children.Remove(nodeControl);
45             }
46             grdMindMap.ColumnDefinitions.Clear();
47             grdMindMap.RowDefinitions.Clear();
48             nodeControls.Clear();
49             DrawNode(mindMap.Root);
50
51         }
52         private int DrawNode(Node node, int level = 0)
53         {
54             TextBlock content = new TextBlock
55             {
56                 Text = node.ToString(),
57                 TextWrapping = TextWrapping.Wrap
58             };
59             UserControl nodeControl = new UserControl
60             {
61                 Content = content,
62                 Tag = node,
63                 BorderBrush = Brushes.Black,
64                 BorderThickness = new Thickness(1),
65                 Background = Brushes.LightGray,
66                 Margin = new Thickness(3),
67                 Cursor = Cursors.Hand,
68                 MinHeight = 50
69             };
70             nodeControl.MouseDoubleClick += NodeControl_Click;
71             nodeControl.TouchUp += NodeControl_Click;
72
73             grdMindMap.Children.Add(nodeControl);
74             nodeControls.Add(nodeControl);
75
76             grdMindMap.RowDefinitions.Add(new RowDefinition());
77         }
78     }
79 }
```

```

78     Grid.SetRow(nodeControl, grdMindMap.RowDefinitions.Count - 1);
79
80     int maxlvl = (level == 0 ? 1 : level);
81     foreach (Node n in node.Childrens)
82     {
83         int nodeMaxLvl = DrawNode(n, level + 1);
84         maxlvl = (maxlvl < nodeMaxLvl ? nodeMaxLvl : maxlvl);
85     }
86
87     while (grdMindMap.ColumnDefinitions.Count <= level)
88     {
89         grdMindMap.ColumnDefinitions.Add(new ColumnDefinition());
90     }
91     Grid.SetColumn(nodeControl, level);
92
93     return maxlvl + 1;
94 }
95
96 private void NodeControl_Click(object sender, EventArgs e)
97 {
98     Node node = (sender as UserControl).Tag as Node;
99     if (node.Previous == null)
100    {
101        MessageBox.Show("Pour changer l'intitulé de noeud, modifiez le mindmap", "Information", MessageBoxButtons.OK);
102    }
103    else
104    {
105        NodeEdit nodeEdit = new NodeEdit(node);
106        if (nodeEdit.ShowDialog() == true)
107        {
108            if (nodeEdit.Deleted)
109            {
110                controller.Delete(node);
111            }
112            else
113            {
114                string name = nodeEdit.tbxName.Text.Trim();
115                Node previous = nodeEdit.cbxPrevious.SelectedItem as Node;
116                controller.UpdateNode(name, previous, node);
117            }
118            Refresh();
119        }
120    }
121 }
122
123 private void Quit_Click(object sender, RoutedEventArgs e)
124 {
125     Close();
126 }
127
128
129 private void Modify_Click(object sender, RoutedEventArgs e)
130 {
131     MindmapEdit mindMapEdit = new MindmapEdit(mindMap);
132     if (mindMapEdit.ShowDialog() == true)
133     {
134         if (mindMapEdit.Deleted)
135         {
136             controller.Delete(mindMap);
137             DialogResult = false;
138             Close();
139         }
140         else
141         {
142             string name = mindMapEdit.tbxName.Text.Trim();
143             controller.UpdateMindMap(name, mindMap);
144         }
145         Refresh();
146     }
147 }
148
149 private void CreateNode_Click(object sender, EventArgs e)
150 {
151     NodeCreate nodeCreate = new NodeCreate(mindMap);
152     if (nodeCreate.ShowDialog() == true)
153     {
154         string name = nodeCreate.tbxName.Text.Trim();
155         Node previous = nodeCreate.cbxPrevious.SelectedItem as Node;

```

```

156         controller.CreateNode(name, previous, mindMap);
157     }
158 }
159 }
160 }
161 }
162 }

```

Listing 35 – ../../Scrum'o'wall/Scrum'o'wall/Views/MindmapMenu.xaml.cs

2.33 NodeCreate.xaml

```

1 <!--
2 * Author      :    Ga l Mariot
3 * Project     :    Scrum'o'wall
4 * File        :    NodeCreate.xaml
5 * Desc.       :    This file contains the basic template of the NodeCreate view
6 -->
7 <Window x:Class="Scrum_o_wall.Views.NodeCreate"
8     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
9     xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
10    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
11    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
12    xmlns:local="clr-namespace:Scrum_o_wall.Views"
13    mc:Ignorable="d"
14    Title="Cr ation_d'un_noeud" Height="240.027" Width="490.14" ↵
15        WindowStartupLocation="CenterScreen" FontSize="16" >
16     <Grid>
17         <Grid.ColumnDefinitions>
18             <ColumnDefinition Width="161" /></ColumnDefinition>
19             <ColumnDefinition /></ColumnDefinition>
20         </Grid.ColumnDefinitions>
21         <Grid.RowDefinitions>
22             <RowDefinition Height="47*" /></RowDefinition>
23             <RowDefinition Height="50*" /></RowDefinition>
24             <RowDefinition Height="62*" /></RowDefinition>
25             <RowDefinition Height="62*" /></RowDefinition>
26         </Grid.RowDefinitions>
27         <TextBlock Grid.ColumnSpan="2" TextAlignment="Center" FontSize="20" ↵
28             VerticalAlignment="Center" Height="28" Margin="0,10,0,9">Cr ation ↵
29                 d'un noeud</TextBlock>
30         <TextBlock Grid.Row="1" TextAlignment="Right" ↵
31             VerticalAlignment="Center" Margin="10">Nom :</TextBlock>
32         <TextBox x:Name="tbxName" Grid.Row="1" MaxLength="100" Grid.Column="1" ↵
33             Margin="10" Stylus.IsPressAndHoldEnabled="False" /></TextBox>
34
35         <TextBlock Grid.Row="2" TextAlignment="Right" ↵
36             VerticalAlignment="Center" Margin="10">Pr c dent :</TextBlock>
37         <ComboBox x:Name="cbxPrevious" Grid.Row="2" Grid.Column="1" Margin="10" ↵
38             Stylus.IsPressAndHoldEnabled="False" /></ComboBox>
39
40         <Grid Grid.Row="3" Grid.ColumnSpan="2">
41             <Grid.ColumnDefinitions>
42                 <ColumnDefinition /></ColumnDefinition>
43                 <ColumnDefinition Width="2*" /></ColumnDefinition>
44             </Grid.ColumnDefinitions>
45             <Button TouchUp="BtnCancel_Click" x:Name="btnCancel" ↵
46                 Grid.Column="0" Margin="10" ↵
47                 Click="BtnCancel_Click">Annuler</Button>
48             <Button TouchUp="BtnConfirm_Click" x:Name="btnConfirm" ↵
49                 Grid.Column="2" Margin="10" ↵
50                 Click="BtnConfirm_Click">Confirmer</Button>
51         </Grid>
52     </Grid>
53 </Window>

```

Listing 36 – ../../Scrum'o'wall/Scrum'o'wall/Views/NodeCreate.xaml

2.34 NodeCreate.xaml.cs

```

1 /*
2 * Author      :    Ga l Serge Mariot
3 * Project     :    Scrum'o'wall
4 * File        :    NodeCreate.xaml.cs

```

```

5 * Desc.      : This file contains the logic in the NodeCreate view
6 */
7 using Scrum_o_wall.Classes;
8 using System;
9 using System.Windows;
10
11 namespace Scrum_o_wall.Views
12 {
13     /// <summary>
14     /// Logique d'interaction pour NodeCreate.xaml
15     /// </summary>
16     public partial class NodeCreate : Window
17     {
18         public NodeCreate(MindMap mindMap)
19         {
20             InitializeComponent();
21             foreach (Node node in mindMap.GetAllNodes())
22             {
23                 cbxPrevious.Items.Add(node);
24             }
25         }
26         private void btnCancel_Click(object sender, EventArgs e)
27         {
28             Close();
29         }
30         private void btnConfirm_Click(object sender, EventArgs e)
31         {
32             if (txbName.Text.Trim().Length > 0 && cbxPrevious.SelectedItem != null)
33             {
34                 DialogResult = true;
35                 Close();
36             }
37             else
38             {
39                 MessageBox.Show("Un ou plusieurs champ(s) n'est pas rempli !", ←
40                             "Erreur", MessageBoxButtons.OK, MessageBoxIcon.Error);
41             }
42         }
43     }
44 }
```

Listing 37 – ../../Scrum'o'wall/Scrum'o'wall/Views/NodeCreate.xaml.cs

2.35 NodeEdit.xaml

```

1 <!--
2 * Author    : Gaël Mariot
3 * Project   : Scrum'o'wall
4 * File      : NodeEdit.xaml
5 * Desc.     : This file contains the basic template of the NodeEdit view
6 -->
7 <Window x:Class="Scrum_o_wall.Views.NodeEdit"
8     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
9     xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
10    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
11    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
12    xmlns:local="clr-namespace:Scrum_o_wall.Views"
13    mc:Ignorable="d"
14    Title="Modification d'un noeud" Height="240.027" Width="490.14" ←
15        WindowStartupLocation="CenterScreen" FontSize="16">
16     <Grid>
17         <Grid.ColumnDefinitions>
18             <ColumnDefinition Width="161" /></ColumnDefinition>
19             <ColumnDefinition /></ColumnDefinition>
20         </Grid.ColumnDefinitions>
21         <Grid.RowDefinitions>
22             <RowDefinition Height="47*" /></RowDefinition>
23             <RowDefinition Height="50*" /></RowDefinition>
24             <RowDefinition Height="62*" /></RowDefinition>
25             <RowDefinition Height="62*" /></RowDefinition>
26         </Grid.RowDefinitions>
27         <TextBlock Grid.ColumnSpan="2" TextAlignment="Center" FontSize="20" ←
28             VerticalAlignment="Center" Height="28" ←
29             Margin="0,10,0,9">Modification d'un noeud</TextBlock>
```

```

27     <TextBlock Grid.Row="1" TextAlignment="Right" ←
28         VerticalAlignment="Center" Margin="10">Nom :</TextBlock>
29     <TextBox x:Name="tbxName" Grid.Row="1" MaxLength="100" Grid.Column="1" ←
30         Margin="10" Stylus.IsPressAndHoldEnabled="False"></TextBox>
31
32     <TextBlock Grid.Row="2" TextAlignment="Right" ←
33         VerticalAlignment="Center" Margin="10">Précédent :</TextBlock>
34     <ComboBox x:Name="cbxPrevious" Grid.Row="2" Grid.Column="1" Margin="10" ←
35         Stylus.IsPressAndHoldEnabled="False"></ComboBox>
36
37     <Grid Grid.Row="3" Grid.ColumnSpan="2">
38         <Grid.ColumnDefinitions>
39             <ColumnDefinition></ColumnDefinition>
40             <ColumnDefinition></ColumnDefinition>
41             <ColumnDefinition Width="2*"></ColumnDefinition>
42         </Grid.ColumnDefinitions>
43         <Button TouchUp="BtnCancel_Click" x:Name="btnCancel" ←
44             Grid.Column="0" Margin="10" ←
45             Click="BtnCancel_Click">Annuler</Button>
46         <Button TouchUp="BtnDelete_Click" x:Name="btnDelete" ←
47             Grid.Column="1" Margin="10" ←
48             Click="BtnDelete_Click">Supprimer</Button>
49         <Button TouchUp="BtnConfirm_Click" x:Name="btnConfirm" ←
50             Grid.Column="2" Margin="10" ←
51             Click="BtnConfirm_Click">Confirmer</Button>
52     </Grid>
53 </Grid>
54 </Window>

```

Listing 38 – ../../Scrum'o'wall/Scrum'o'wall/Views/NodeEdit.xaml

2.36 NodeEdit.xaml.cs

```

1 /*
2 * Author : Gael Serge Mariot
3 * Project : Scrum'o'wall
4 * File : NodeEdit.xaml.cs
5 * Desc. : This file contains the logic in the NodeEdit view
6 */
7 using Scrum_o_wall.Classes;
8 using System;
9 using System.Windows;
10
11 namespace Scrum_o_wall.Views
12 {
13     /// <summary>
14     /// Logique d'interaction pour NodeCreate.xaml
15     /// </summary>
16     public partial class NodeEdit : Window
17     {
18         public bool Deleted = false;
19         private readonly Node node;
20         public NodeEdit(Node aNode)
21         {
22             node = aNode;
23             InitializeComponent();
24             tbxName.Text = node.Name;
25             if (aNode.Previous != null)
26             {
27                 foreach (Node n in node.MindMap.GetAllNodes())
28                 {
29                     cbxPrevious.Items.Add(n);
30                 }
31                 cbxPrevious.SelectedItem = aNode.Previous;
32             }
33         }
34         private void BtnCancel_Click(object sender, EventArgs e)
35         {
36             Close();
37         }
38         private void BtnConfirm_Click(object sender, EventArgs e)
39         {
40             if (tbxName.Text.Trim().Length > 0 && cbxPrevious.SelectedItem != null)
41             {
42                 DialogResult = true;
43             }
44         }
45     }
46 }

```

```

44             Close();
45         }
46     else
47     {
48         MessageBox.Show("Un ou plusieurs champ(s) n'est pas rempli!", ←
49                         "Erreur", MessageBoxButtons.OK, MessageBoxIcon.Error);
50     }
51 }
52 private void BtnDelete_Click(object sender, EventArgs e)
53 {
54     if (MessageBox.Show("Le noeud sera supprimé.\nêtes-vous sûr(e)?", ←
55                         "Attention", MessageBoxButtons.YesNo, MessageBoxIcon.Warning) == ←
56                         DialogResult.Yes)
57     {
58         Deleted = true;
59         DialogResult = true;
60         Close();
61     }
}

```

Listing 39 – ../../Scrum'o'wall/Scrum'o'wall/Views/NodeEdit.xaml.cs

2.37 ProjectCreate.xaml

```

1 <!--
2 * Author    :   Gael Mariot
3 * Project   :   Scrum'o'wall
4 * File      :   ProjectCreate.xaml
5 * Desc.     :   This file contains the basic template of the ProjectCreate view
6 -->
7 <Window x:Class="Scrum_o_wall.Views.ProjectCreate"
8     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
9     xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
10    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
11    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
12    xmlns:local="clr-namespace:Scrum_o_wall.Views"
13    mc:Ignorable="d"
14    Title="Création d'un projet" Height="500" Width="600" ←
15        WindowStartupLocation="CenterScreen" FontSize="16" >
16     <Grid Margin="0,0,2,-1">
17         <Grid.ColumnDefinitions>
18             <ColumnDefinition Width="199"/>
19             <ColumnDefinition/>
20         </Grid.ColumnDefinitions>
21         <Grid.RowDefinitions>
22             <RowDefinition Height="94*"/>
23             <RowDefinition Height="54*"/>
24             <RowDefinition Height="200*"/>
25             <RowDefinition Height="59*"/>
26             <RowDefinition Height="63*"/>
27         </Grid.RowDefinitions>
28         <Label HorizontalAlignment="Center" VerticalAlignment="Center" ←
29             FontSize="36" Height="58" Margin="172,18,168,18" Width="250" ←
30             Grid.ColumnSpan="2">Créer un projet</Label>
31
32         <Label Grid.Row="1" HorizontalAlignment="Right" ←
33             VerticalAlignment="Center" Margin="10">Nom du projet :</Label>
34         <TextBox Grid.Row="1" Name="tbxName" VerticalAlignment="Center" ←
35             Grid.Column="1" Margin="10" MaxLength="50" ←
36             Stylus.IsPressAndHoldEnabled="False"></TextBox>
37
38         <Label Grid.Row="2" HorizontalAlignment="Right" Margin="10">Description ←
39             du projet :</Label>

```

```

40         <Button TouchUp="BtnAddProject_Click" x:Name="btnAddProject" ↵
41             Click="BtnAddProject_Click" Grid.Row="4" Grid.Column="1" ↵
42             Margin="10">Confirmer</Button>
43     </Grid>
44 </Window>

```

Listing 40 – ../../Scrum'o'wall/Scrum'o'wall/Views/ProjectCreate.xaml

2.38 ProjectCreate.xaml.cs

```

1 /*
2 * Author : Ga l Serge Mariot
3 * Project : Scrum'o'wall
4 * File : ProjectCreate.xaml.cs
5 * Desc. : This file contains the logic in the ProjectCreate view
6 */
7 using System.Windows;
8
9 namespace Scrum_o_wall.Views
10 {
11     /// <summary>
12     /// Logique d'interaction pour Window1.xaml
13     /// </summary>
14     public partial class ProjectCreate : Window
15     {
16         public ProjectCreate()
17         {
18             InitializeComponent();
19         }
20
21         private void btnCancel_Click(object sender, RoutedEventArgs e)
22         {
23
24             Close();
25         }
26         private void BtnAddProject_Click(object sender, RoutedEventArgs e)
27         {
28             int nameLength = tbxName.Text.Trim().Length;
29             int descLength = tbxDesc.Text.Trim().Length;
30             if (nameLength > 0 && descLength > 0 && tbxDate.SelectedDate != null)
31             {
32                 DialogResult = true;
33                 Close();
34             }
35             else
36             {
37                 MessageBox.Show("Un ou plusieurs champ(s) n'est pas rempli !", ←
38                             "Erreur", MessageBoxButton.OK, MessageBoxImage.Error);
39             }
40         }
41     }

```

Listing 41 – ../../Scrum'o'wall/Scrum'o'wall/Views/ProjectCreate.xaml.cs

2.39 ProjectEdit.xaml

```

1 <!--
2 * Author : Ga l Mariot
3 * Project : Scrum'o'wall
4 * File : ProjectEdit.xaml
5 * Desc. : This file contains the basic template of the ProjectEdit view
6 -->
7 <Window x:Class="Scrum_o_wall.Views.ProjectEdit"
8     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
9     xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
10    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
11    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
12    xmlns:local="clr-namespace:Scrum_o_wall.Views"
13    mc:Ignorable="d"
14    Title="Modification d'un projet" Height="500" Width="600" ←
15        WindowStartupLocation="CenterScreen" FontSize="16" >
16        <Grid Margin="0,0,2,-1">
17            <Grid.ColumnDefinitions>

```

```

17     <ColumnDefinition Width="140*"/>
18     <ColumnDefinition Width="249*"/>
19 </Grid.ColumnDefinitions>
20 <Grid.RowDefinitions>
21     <RowDefinition Height="84*"/>
22     <RowDefinition Height="43*"/>
23     <RowDefinition Height="184*"/>
24     <RowDefinition Height="53*"/>
25     <RowDefinition Height="56*"/>
26     <RowDefinition Height="56*"/>
27 </Grid.RowDefinitions>
28 <Label Grid.ColumnSpan="2" HorizontalAlignment="Center" ←
    VerticalAlignment="Center" FontSize="36"   >Modification un ←
    projet</Label>
29
30 <Label Grid.Row="1" HorizontalAlignment="Right" ←
    VerticalAlignment="Center">Nom du projet :</Label>
31 <TextBox Grid.Row="1" Name="txbName" Grid.Column="1" MaxLength="50" ←
    Margin="10"/>
32
33 <Label Grid.Row="2" HorizontalAlignment="Right" ←
    VerticalAlignment="Top">Description du projet :</Label>
34 <TextBox Grid.Row="2" Name="txbDesc" Grid.Column="1" MaxLength="65535" ←
    TextWrapping="WrapWithOverflow" Margin="10" ←
    Stylus.IsPressAndHoldEnabled="False"/>
35
36 <Label Grid.Row="3" HorizontalAlignment="Right" ←
    VerticalAlignment="Center">Date de début :</Label>
37 <DatePicker Grid.Row="3" Name="dtpckrDateBegin" Grid.Column="1" ←
    Margin="10" VerticalAlignment="Center" ←
    Stylus.IsPressAndHoldEnabled="False"/>
38
39 <Grid Grid.Row="4" Grid.ColumnSpan="2">
40     <Grid.ColumnDefinitions>
41         <ColumnDefinition></ColumnDefinition>
42         <ColumnDefinition></ColumnDefinition>
43     </Grid.ColumnDefinitions>
44     <Button TouchUp="BtnStates_Click" Margin="10" Name="btnStates" ←
        Click="BtnStates_Click">Colonnes</Button>
45     <Button TouchUp="BtnUsers_Click" Grid.Column="1" Margin="10" ←
        Name="btnUsers" Click="BtnUsers_Click">Utilisateurs ←
        assignés</Button>
46 </Grid>
47 <Grid Grid.Row="5" Grid.ColumnSpan="2">
48     <Grid.ColumnDefinitions>
49         <ColumnDefinition></ColumnDefinition>
50         <ColumnDefinition></ColumnDefinition>
51         <ColumnDefinition Width="2*"></ColumnDefinition>
52     </Grid.ColumnDefinitions>
53     <Button x:Name="btnCancel" Margin="10" Click="BtnCancel_Click" ←
        TouchUp="BtnCancel_Click">Annuler</Button>
54     <Button x:Name="btnDelete" Grid.Column="1" Margin="10" ←
        Click="BtnDelete_Click" ←
        TouchUp="BtnDelete_Click">Supprimer</Button>
55     <Button x:Name="btnConfirm" Grid.Column="2" Margin="10" ←
        Click="BtnConfirm_Click" ←
        TouchUp="BtnConfirm_Click">Confirmer</Button>
56 </Grid>
57 </Grid>
58 </Window>

```

Listing 42 – .../Scrum'o'wall/Scrum'o'wall/Views/ProjectEdit.xaml

2.40 ProjectEdit.xaml.cs

```

1 /*
2 * Author      :  Gael Serge Mariot
3 * Project     :  Scrum'o'wall
4 * File        :  ProjectEdit.xaml.cs
5 * Desc.       :  This file contains the logic in the ProjectEdit view
6 */
7 using Scrum_o_wall.Classes;
8 using System;
9 using System.Windows;
10
11 namespace Scrum_o_wall.Views
12 {

```

```

13 /// <summary>
14 /// Logique d'interaction pour ProjectEdit.xaml
15 /// </summary>
16 public partial class ProjectEdit : Window
17 {
18     private readonly Project project;
19     public bool Deleted = false;
20     private readonly Controller controller;
21     public ProjectEdit(Project aProject, Controller aController)
22     {
23         project = aProject;
24         controller = aController;
25
26         InitializeComponent();
27
28         tbxDesc.Text = project.Description;
29         tbxName.Text = project.Name;
30         dtpckrDateBegin.SelectedDate = project.Begin;
31     }
32
33     private void BtnConfirm_Click(object sender, EventArgs e)
34     {
35         int nameLength = tbxName.Text.Trim().Length;
36         int descLength = tbxDesc.Text.Trim().Length;
37         if (dtpckrDateBegin.SelectedDate != null && descLength > 0 && ←
38             nameLength > 0)
39         {
40             DialogResult = true;
41             Close();
42         }
43         else
44         {
45             MessageBox.Show("Un ou plusieurs champ(s) n'est pas rempli!", ←
46                             "Erreur", MessageBoxButtons.OK, MessageBoxIcon.Error);
47         }
48     }
49     private void BtnStates_Click(object sender, EventArgs e)
50     {
51         StateMenu stateMenu = new StateMenu(project, controller);
52         stateMenu.ShowDialog();
53     }
54     private void BtnUsers_Click(object sender, EventArgs e)
55     {
56         UserMenu userMenu = new UserMenu(project, controller.Users, ←
57                                         controller);
58         userMenu.ShowDialog();
59     }
60     private void btnCancel_Click(object sender, EventArgs e)
61     {
62         Close();
63     }
64     private void BtnDelete_Click(object sender, EventArgs e)
65     {
66         if (MessageBox.Show("Le projet sera supprimé.\nVoulez-vous vraiment?", "Attention", MessageBoxButtons.YesNo, MessageBoxIcon.Warning) == DialogResult.Yes)
67         {
68             DialogResult = true;
69             Deleted = true;
70             Close();
71         }
72     }
73 }

```

Listing 43 – ../../Scrum'o'wall/Scrum'o'wall/Views/ProjectEdit.xaml.cs

2.41 ProjectMenu.xaml

```

1 <!--
2 * Author    :   Gaël Mariot
3 * Project   :   Scrum'o'wall
4 * File      :   ProjectMenu.xaml
5 * Desc.     :   This file contains the basic template of the ProjectMenu view
6 -->

```

```

7 <Window x:Class="Scrum_o_wall.Views.ProjectMenu"
8   xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
9   xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
10  xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
11  xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
12  xmlns:local="clr-namespace:Scrum_o_wall.Views"
13  mc:Ignorable="d"
14  Title="Scrum'o'Wall" Height="450" Width="800" WindowState="Maximized" ←
15    WindowStyle="None" FontSize="16" ←
16      WindowStartupLocation="CenterScreen" ResizeMode="NoResize">
17  <Window.ContextMenu >
18    <ContextMenu >
19      <MenuItem Header="Editer le projet" Name="Edit" Click="Edit_Click"/>
20      <Separator/>
21      <MenuItem Header="Retour" Name="Quit" Click="Quit_Click"/>
22  </ContextMenu>
23  <Canvas x:Name="cnvsBacklog">
24    <Label Name="lblProjectName" Content="Nom du projet" FontSize="30" ←
25      Height="60"/>
26    <Label Name="lblBacklog" Canvas.Top="65" Height="30">Product ←
27      Backlog</Label>
28    <GroupBox x:Name="gbxUserStories" Height="260" Width="280" ←
29      Canvas.Top="100" Canvas.Left="10" Header="User Stories" >
30      <ScrollViewer Name="scrlUserStories" Margin="0,0,0,0">
31        <Canvas x:Name="cnvsUserStories">
32
33          </Canvas>
34        </ScrollViewer>
35      </GroupBox>
36    <GroupBox x:Name="gbxSprints" Height="260" Width="220" Canvas.Top="100" ←
37      Canvas.Right="10" Canvas.Left="295" Header="Sprints" >
38      <ScrollViewer Name="scrlSprints" Margin="0,0,0,0">
39        <Canvas x:Name="cnvsSprints">
40
41          </Canvas>
42        </ScrollViewer>
43      </GroupBox>
44    <GroupBox x:Name="gbxMindMap" Height="260" Width="270" Canvas.Top="100" ←
45      Canvas.Right="10" Canvas.Left="520" Header="MindMaps" >
46      <ScrollViewer Name="scrlMindMaps" Margin="0,0,0,0">
47        <Canvas x:Name="cnvsMindMaps">
48
49          </Canvas>
50        </ScrollViewer>
51      </GroupBox>
52    <Button x:Name="btnReturn" TouchUp="Quit_Click" Click="Quit_Click" ←
53      Content="à " VerticalContentAlignment="Top" Canvas.Bottom="10" ←
54      Width="75" Height="75" BorderBrush="Black" FontSize="45" ←
55      Canvas.Top="365" Canvas.Left="363">
56      <Button.Resources>
57        <Style TargetType="Border">
58          <Setter Property="CornerRadius" Value="50"/>
59        </Style>
60      </Button.Resources>
61    </Button>
62    <Button x:Name="btnAddUserStory" TouchUp="BtnAddUserStory_Click" ←
63      Click="BtnAddUserStory_Click" Content="+" ←
64      VerticalContentAlignment="Top" Canvas.Bottom="10" Width="40" ←
65      Height="40" BorderBrush="Black" FontSize="23" Canvas.Top="320" ←
66      Canvas.Left="193">
67      <Button.Resources>
68        <Style TargetType="{x:Type Border}">
69          <Setter Property="CornerRadius" Value="50"/>
70        </Style>
71      </Button.Resources>
72    </Button>
73    <Button x:Name="btnAddSprint" TouchUp="BtnAddSprint_Click" ←
74      Click="BtnAddSprint_Click" Content="+" ←
75      VerticalContentAlignment="Top" Canvas.Bottom="10" Width="40" ←
76      Height="40" BorderBrush="Black" FontSize="23" Canvas.Top="320" ←
77      Canvas.Left="380">
78      <Button.Resources>
79        <Style TargetType="{x:Type Border}">
80          <Setter Property="CornerRadius" Value="50"/>
81        </Style>
82      </Button.Resources>
83    </Button>

```

```

67         <Button x:Name="btnAddMindMap" TouchUp="BtnAddMindMap_Click" ↵
68             Click="BtnAddMindMap_Click" Content="+" ↵
69             VerticalContentAlignment="Top" Canvas.Bottom="10" Width="40" ↵
70             Height="40" BorderBrush="Black" FontSize="23" Canvas.Top="333" ↵
71             Canvas.Left="656">
72             <Button.Resources>
73                 <Style TargetType="{x:Type Border}">
74                     <Setter Property="CornerRadius" Value="50"/>
75                 </Style>
76             </Button.Resources>
77         </Button>
78     </Canvas>
79 </Window>

```

Listing 44 – ../../Scrum'o'wall/Scrum'o'wall/Views/ProjectMenu.xaml

2.42 ProjectMenu.xaml.cs

```

/*
 * Author    :   Ga l Serge Mariot
 * Project   :   Scrum'o'wall
 * File      :   ProjectMenu.xaml.cs
 * Desc.     :   This file contains the logic in the ProjectMenu view
*/
using Scrum_o_wall.Classes;
using System;
using System.Collections.Generic;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Input;
using System.Windows.Media;
namespace Scrum_o_wall.Views
{
    /// <summary>
    /// Logique d'interaction pour ProjectMenu.xaml
    /// </summary>
    public partial class ProjectMenu : Window
    {
        private readonly Project project;
        private readonly Controller controller;
        private List<UserControl> sprintUserControls = new List<UserControl>();
        private List<UserControl> userStoriesControls = new List<UserControl>();
        private List<UserControl> mindMapControls = new List<UserControl>();

        private Dictionary<InputDevice, Point> currentPoint = new ←
            Dictionary<InputDevice, Point>();
        private Dictionary<InputDevice, UserControl> infos = new ←
            Dictionary<InputDevice, UserControl>();

        public ProjectMenu(Project aProject, Controller aController)
        {
            InitializeComponent();
            project = aProject;
            controller = aController;

            lblProjectName.Content = aProject.Name;

            Loaded += ProjectMenu_Loaded;
            PreviewTouchMove += Canvas_PreviewTouchMove;
            TouchUp += Canvas_TouchUp;
        }

        private void Canvas_TouchUp(object sender, TouchEventArgs e)
        {
            if (currentPoint.ContainsKey(e.Device))
            {
                currentPoint.Remove(e.Device);
                infos.Remove(e.Device);
            }
        }

        private void Refresh()
        {

```

```

58     int nbMindMaps = project.MindMaps.Count;
59     foreach (UserControl mindmapControl in mindMapControls)
60     {
61         cnvsMindMaps.Children.Remove(mindmapControl);
62     }
63     mindMapControls.Clear();
64     for (int i = 0; i < nbMindMaps; i++)
65     {
66         MindMap mindMap = project.MindMaps[i];
67         //Create Sprint frame
68         UserControl mindmapControl = new UserControl
69         {
70             Content = mindMap.ToString(),
71             Width = cnvsMindMaps.Width - 40,
72             BorderThickness = new Thickness(1),
73             BorderBrush = Brushes.Black,
74             Cursor = Cursors.Hand,
75             Height = 50,
76             Tag = mindMap
77         };
78         mindmapControl.MouseDoubleClick += MindmapControl_Click;
79         mindmapControl.TouchDown += MindmapControl_Click;
80
81         cnvsMindMaps.Children.Add(mindmapControl);
82         mindMapControls.Add(mindmapControl);
83
84         Canvas.SetLeft(mindmapControl, 5);
85         Canvas.SetTop(mindmapControl, 60 * i);
86     }
87
88     int nbSprints = project.Sprints.Count;
89     foreach (UserControl sprintControl in sprintUserControls)
90     {
91         cnvsSprints.Children.Remove(sprintControl);
92     }
93     sprintUserControls.Clear();
94     for (int i = 0; i < nbSprints; i++)
95     {
96         Sprint sprint = project.Sprints[i];
97         //Create Sprint frame
98         UserControl sprintControl = new UserControl
99         {
100             Content = sprint.ToString(),
101             Width = cnvsSprints.Width - 40,
102             BorderThickness = new Thickness(1),
103             BorderBrush = Brushes.Black,
104             Cursor = Cursors.Hand,
105             Height = 50,
106             Tag = sprint
107         };
108         sprintControl.MouseDoubleClick += Sprint_Click;
109         sprintControl.TouchDown += Sprint_Click;
110         sprintControl.PreviewTouchUp += Sprint_PreviewTouchUp;
111         sprintControl.TouchEnter += Sprint_TouchEnter;
112         sprintControl.TouchLeave += Sprint_TouchLeave;
113         //Change Color by DateRange
114         if (DateTime.Now > sprint.Begin && DateTime.Now < sprint.End)
115         {
116             //Actual
117             sprintControl.Background = Brushes.LightBlue;
118         }
119         else if (DateTime.Now < sprint.Begin)
120         {
121             //Not beginned
122             sprintControl.Background = Brushes.LightGray;
123         }
124         else
125         {
126             //Already passed
127             sprintControl.Background = Brushes.LightPink;
128         }
129
130         cnvsSprints.Children.Add(sprintControl);
131         sprintUserControls.Add(sprintControl);
132
133         Canvas.SetLeft(sprintControl, 5);
134         Canvas.SetTop(sprintControl, 60 * i);
135     }
136 }
```

```

137
138     int nbUserStories = project.AllUserStories.Count;
139     foreach (UserControl userStoryControl in userStoriesControls)
140     {
141         cnvsUserStories.Children.Remove(userStoryControl);
142     }
143     userStoriesControls.Clear();
144     for (int i = 0; i < nbUserStories; i++)
145     {
146         UserStory userStory = project.AllUserStories[i];
147         //Create userStory frame
148         TextBlock content = new TextBlock
149         {
150             Text = userStory.ToString(),
151             TextWrapping = TextWrapping.Wrap
152         };
153         UserControl userStoryControl = new UserControl
154         {
155             Content = content,
156             Cursor = Cursors.Hand,
157             Height = 50,
158             Tag = userStory,
159             Width = cnvsUserStories.Width - 40,
160             BorderBrush = Brushes.Black,
161             BorderThickness = new Thickness(1)
162         };
163         // Change color by limit date (if exists and happened or not)
164         if (userStory.DateLimit != null)
165         {
166             if (DateTime.Now > userStory.DateLimit)
167             {
168                 userStoryControl.Background = Brushes.LightPink;
169             }
170             else
171             {
172                 userStoryControl.Background = Brushes.LightBlue;
173             }
174         }
175         else
176         {
177             userStoryControl.Background = Brushes.LightGray;
178         }
179         userStoryControl.MouseDoubleClick += UserStory_MouseDoubleClick;
180         userStoryControl.PreviewMouseDown += UserStory_PreviewMouseDown;
181         userStoryControl.TouchUp += UsrCtrlUserStory_TouchUp;
182
183         Stylus.SetIsPressAndHoldEnabled(userStoryControl, false);
184
185         cnvsUserStories.Children.Add(userStoryControl);
186         userStoriesControls.Add(userStoryControl);
187
188         Canvas.SetLeft(userStoryControl, 5);
189         Canvas.SetTop(userStoryControl, 60 * i);
190     }
191 }
192
193 private void MindmapControl_Click(object sender, EventArgs e)
194 {
195     MindMap m = (sender as UserControl).Tag as MindMap;
196     MindmapMenu mindmapMenu = new MindmapMenu(m, controller);
197     mindmapMenu.ShowDialog();
198     Refresh();
199 }
200
201 private void Sprint_TouchEnter(object sender, TouchEventArgs e)
202 {
203     if (currentPoint.ContainsKey(e.Device))
204     {
205         (sender as UserControl).BorderThickness = new Thickness(3);
206     }
207 }
208 private void Sprint_TouchLeave(object sender, TouchEventArgs e)
209 {
210     if (currentPoint.ContainsKey(e.Device))
211     {
212         (sender as UserControl).BorderThickness = new Thickness(1);
213     }
214 }
215

```

```

216
217     private void UserStoryEditing(UserStory userStory)
218     {
219         UserStoryEdit userStoryEdit = new UserStoryEdit(userStory, project, ←
220             controller);
221
222         if (userStoryEdit.ShowDialog() == true)
223         {
224             if (userStoryEdit.Deleted)
225             {
226                 controller.Delete(userStory);
227             }
228             else
229             {
230                 string desc = userStoryEdit.tbxDesc.Text.Trim();
231                 DateTime? dateLimite = ←
232                     userStoryEdit.dtpckrDateLimit.SelectedDate;
233                 int complexity = ←
234                     Convert.ToInt32(userStoryEdit.tbxComplexity.Text);
235                 int completedComplexity = ←
236                     Convert.ToInt32(userStoryEdit.tbxCompletedComplexity.Text);
237                 bool blocked = userStoryEdit.chkBxBlocked.IsChecked == true;
238                 Priority priority = ←
239                     (Priority)userStoryEdit.cbxPriority.SelectedItem;
240                 Classes.Type type = ←
241                     (Classes.Type)userStoryEdit.cbxType.SelectedItem;
242                 State state = userStory.State;
243
244                 controller.UpdateUserStory(desc, dateLimite, complexity, ←
245                     completedComplexity, blocked, priority, type, state, ←
246                     userStory);
247             }
248             Refresh();
249         }
250     }
251
252     private void ProjectMenu_Loaded(object sender, RoutedEventArgs e)
253     {
254         //set controls sizes
255         cnvsBacklog.Width = ActualWidth;
256         cnvsBacklog.Height = ActualHeight;
257
258         gbxUserStories.Width = (cnvsBacklog.Width - 25) / 3.0;
259         gbxUserStories.Height = (cnvsBacklog.Height - 250);
260
261         gbxSprints.Width = (cnvsBacklog.Width - 25) / 3.0;
262         gbxSprints.Height = (cnvsBacklog.Height - 250);
263
264         gbxMindMap.Width = (cnvsBacklog.Width - 25) / 3.0;
265         gbxMindMap.Height = (cnvsBacklog.Height - 250);
266
267         cnvsMindMaps.Width = gbxMindMap.Width;
268         cnvsMindMaps.Height = gbxMindMap.Height - 25;
269
270         cnvsSprints.Width = gbxSprints.Width;
271         cnvsSprints.Height = gbxSprints.Height - 25;
272
273         cnvsUserStories.Width = gbxUserStories.Width;
274         cnvsUserStories.Height = gbxUserStories.Height - 25;
275
276         //Set controls positions
277         Canvas.SetLeft(gbxUserStories, 10);
278         Canvas.SetLeft(gbxSprints, gbxUserStories.Width + ←
279             Canvas.GetLeft(gbxUserStories) + 5);
280         Canvas.SetLeft(gbxMindMap, gbxSprints.Width + ←
281             Canvas.GetLeft(gbxSprints) + 5);
282
283         Canvas.SetLeft(lblProjectName, (cnvsBacklog.Width - ←
284             lblProjectName.ActualWidth) / 2.0);
285         Canvas.SetLeft(lblBacklog, (cnvsBacklog.Width - ←
286             lblBacklog.ActualWidth) / 2.0);
287
288         Canvas.SetLeft(btnReturn, (cnvsBacklog.Width - ←
289             btnReturn.ActualWidth) / 2.0);
290         Canvas.SetTop(btnReturn, cnvsBacklog.Height - ←
291             btnReturn.ActualHeight - 10);
292
293         Canvas.SetLeft(btnAddSprint, Canvas.GetLeft(gbxSprints) + ←
294             (gbxSprints.Width - btnAddSprint.ActualWidth) / 2.0);

```

```

280     Canvas.SetTop(btnAddSprint, Canvas.GetTop(gbxSprints) + ←
281                 gbxSprints.Height + 10);
282
283     Canvas.SetLeft(btnAddMindMap, Canvas.GetLeft(gbxMindMap) + ←
284                 (gbxMindMap.Width - btnAddMindMap.ActualWidth) / 2.0);
285     Canvas.SetTop(btnAddMindMap, Canvas.GetTop(gbxMindMap) + ←
286                 gbxMindMap.Height + 10);
287
288     //Refresh the window
289     Refresh();
290 }
291 private void UserStory_MouseDoubleClick(object sender, ←
292                                         MouseEventArgs e)
293 {
294     UserStory userStory = (sender as UserControl).Tag as UserStory;
295     UserStoryEditing(userStory);
296 }
297 private void Sprint_Click(object sender, EventArgs e)
298 {
299     Sprint s = (sender as UserControl).Tag as Sprint;
300     SprintMenu sprintMenu = new SprintMenu(s, controller);
301     sprintMenu.ShowDialog();
302     Refresh();
303 }
304 private void Quit_Click(object sender, EventArgs e)
305 {
306     Close();
307 }
308 private void BtnAddUserStory_Click(object sender, EventArgs e)
309 {
310     UserStoryCreate userStoryCreate = new UserStoryCreate(controller);
311     if (userStoryCreate.ShowDialog() == true)
312     {
313         string desc = userStoryCreate.tbxDesc.Text.Trim();
314         DateTime? dateLimite = ←
315             userStoryCreate.dtpckrDateLimit.SelectedDate;
316         int complexity = ←
317             Convert.ToInt32(userStoryCreate.tbxComplexity.Text);
318         Priority priority = ←
319             (Priority)userStoryCreate.cbxPriority.SelectedItem;
320         Classes.Type type = ←
321             (Classes.Type)userStoryCreate.cbxType.SelectedItem;
322         controller.CreateUserStory(desc, dateLimite, complexity, ←
323             priority, type, project);
324         Refresh();
325     }
326 }
327 private void BtnAddSprint_Click(object sender, EventArgs e)
328 {
329     SprintCreate sprintCreate = new SprintCreate();
330     if (sprintCreate.ShowDialog() == true)
331     {
332         DateTime begin = ←
333             (DateTime)sprintCreate.dtpckDateBegin.SelectedDate;
334         DateTime end = (DateTime)sprintCreate.dtpckDateEnd.SelectedDate;
335         controller.CreateSprint(begin, end, project);
336         Refresh();
337     }
338 }
339 private void BtnAddMindMap_Click(object sender, EventArgs e)
340 {
341     MindmapCreate mindmapCreate = new MindmapCreate();
342     if (mindmapCreate.ShowDialog() == true)
343     {
344         string name = mindmapCreate.tbxName.Text.Trim();
345         controller.CreateMindMap(name, project);
346         Refresh();
347     }
348 }
349 private void Sprint_PreviewTouchUp(object sender, TouchEventArgs e)
350 {
351     if (currentPoint.ContainsKey(e.Device))
352     {

```

```

347         currentPoint[e.Device] = e.GetTouchPoint(null).Position;
348         UserControl sprintControl = sender as UserControl;
349         sprintControl.BorderThickness = new Thickness(1);
350         double leftBound = Canvas.GetLeft(sprintControl) + ←
351             Canvas.GetLeft(gbxSprints);
352         double rightBound = leftBound + sprintControl.Width;
353         double topBound = Canvas.GetTop(sprintControl) + ←
354             Canvas.GetTop(cnvsSprints);
355         double bottomBound = topBound + sprintControl.Height;
356
357         Sprint sprint = sprintControl.Tag as Sprint;
358         UserStory userStory = (infos[e.Device] as UserControl).Tag as ←
359             UserStory;
360
361         if (controller.AddUserStoryToSprint(userStory, sprint))
362         {
363             Task.Factory.StartNew(() => MessageBox.Show("Enregistrement ↵
364                 rajout ", "Confirmation", MessageBoxButton.OK, ←
365                     MessageBoxImage.Information));
366         }
367         else
368         {
369             Task.Factory.StartNew(() => MessageBox.Show("Enregistrement ↵
370                 d j existant", "Erreur", MessageBoxButton.OK, ←
371                     MessageBoxImage.Error));
372         }
373
374         currentPoint.Remove(e.Device);
375         infos.Remove(e.Device);
376     }
377 }
378 private void UserStory_PreviewMouseDown(object sender, TouchEventArgs e)
379 {
380     if (currentPoint.ContainsKey(e.Device) || infos.ContainsKey(e.Device))
381     {
382         currentPoint.Remove(e.Device);
383         infos.Remove(e.Device);
384     }
385     currentPoint.Add(e.Device, e.GetTouchPoint(null).Position);
386     infos.Add(e.Device, sender as UserControl);
387 }
388 private void Canvas_PreviewMouseMove(object sender, TouchEventArgs e)
389 {
390     if (currentPoint.ContainsKey(e.Device))
391     {
392         currentPoint[e.Device] = e.GetTouchPoint(null).Position;
393     }
394 }
395 private void UsrCtrlUserStory_TouchUp(object sender, TouchEventArgs e)
396 {
397     if (currentPoint.ContainsKey(e.Device))
398     {
399         currentPoint.Remove(e.Device);
400         infos.Remove(e.Device);
401     }
402     else
403     {
404         UserStory userStory = (sender as UserControl).Tag as UserStory;
405         UserStoryEditing(userStory);
406     }
407 }
408 private void Edit_Click(object sender, RoutedEventArgs e)
409 {
410     ProjectEdit projectEdit = new ProjectEdit(project, controller);
411     if (projectEdit.ShowDialog() == true)
412     {
413         if (projectEdit.Deleted)
414         {
415             controller.Delete(project);
416             DialogResult = true;
417             Close();
418         }
419         else
420         {
421             string name = projectEdit.tbxName.Text.Trim();
422             string desc = projectEdit.tbxDesc.Text.Trim();
423             DateTime dateBegin = ←
424                 (DateTime)projectEdit.dtpckrDateBegin.SelectedDate;
425             controller.UpdateProject(name, desc, dateBegin, project);
426         }
427     }
428 }

```

```

418             Refresh();
419         }
420     }
421 }
422 }
423 }
424 }

```

Listing 45 – ../../Scrum'o'wall/Scrum'o'wall/Views/ProjectMenu.xaml.cs

2.43 SprintCreate.xaml

```

1 <!--
2 * Author      :    Ga l Mariot
3 * Project     :    Scrum'o'wall
4 * File        :    SprintCreate.xaml
5 * Desc.       :    This file contains the basic template of the SprintCreate view
6 -->
7 <Window x:Class="Scrum_o_wall.Views.SprintCreate"
8     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
9     xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
10    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
11    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
12    xmlns:local="clr-namespace:Scrum_o_wall.Views"
13    mc:Ignorable="d"
14    Title="Cr ation_d'un_sprint" Height="281.121" Width="462.967" ↫
15        WindowStartupLocation="CenterScreen"   FontSize="16" >
16     <Grid>
17         <Grid.ColumnDefinitions>
18             <ColumnDefinition Width="140*"/>
19             <ColumnDefinition Width="249*"/>
20         </Grid.ColumnDefinitions>
21         <Grid.RowDefinitions>
22             <RowDefinition Height="84*"/>
23             <RowDefinition Height="43*"/>
24             <RowDefinition Height="43*"/>
25             <RowDefinition Height="53*"/>
26         </Grid.RowDefinitions>
27         <Label Grid.ColumnSpan="2" HorizontalAlignment="Center" ↫
28             VerticalAlignment="Center" FontSize="36" Margin="10">Cr er un ↫
29             sprint</Label>
30         <Label Grid.Row="1" HorizontalAlignment="Right" ↫
31             VerticalAlignment="Center" >Date de d but :</Label>
32         <Label Grid.Row="2" HorizontalAlignment="Right" ↫
33             VerticalAlignment="Center" >Date de fin :</Label>
34         <Button TouchUp="BtnCancel_Click" x:Name="btnCancel" ↫
35             Click="BtnCancel_Click" Grid.Row="3" Margin="10">Annuler</Button>
36
37         <DatePicker Grid.Row="1" Name="dtpckDateBegin" Grid.Column="1" ↫
38             Margin="10" Stylus.IsPressAndHoldEnabled="False"/>
39         <DatePicker Grid.Row="2" Name="dtpckDateEnd" Grid.Column="1" ↫
40             Margin="10" Stylus.IsPressAndHoldEnabled="False"/>
41         <Button TouchUp="BtnAddProject_Click" x:Name="btnAddProject" ↫
42             Click="BtnAddProject_Click" Grid.Row="3" Grid.Column="1" ↫
43             Margin="10">Confirmer</Button>
44     </Grid>
45 </Window>

```

Listing 46 – ../../Scrum'o'wall/Scrum'o'wall/Views/SprintCreate.xaml

2.44 SprintCreate.xaml.cs

```

1 /*
2 * Author      :    Ga l Serge Mariot
3 * Project     :    Scrum'o'wall
4 * File        :    SprintCreate.xaml.cs
5 * Desc.       :    This file contains the logic in the SprintCreate view
6 */
7 using System;
8 using System.Windows;
9
10 namespace Scrum_o_wall.Views
11 {
12     /// <summary>

```

```

13 /// Logique d'interaction pour SprintCreate.xaml
14 /// </summary>
15 public partial class SprintCreate : Window
16 {
17     public SprintCreate()
18     {
19         InitializeComponent();
20     }
21
22     private void BtnCancel_Click(object sender, EventArgs e)
23     {
24
25         Close();
26     }
27     private void BtnAddProject_Click(object sender, EventArgs e)
28     {
29         if (dtpckDateBegin.SelectedDate != null && ←
30             dtpckDateEnd.SelectedDate != null)
31         {
32             DialogResult = true;
33             Close();
34         }
35         else
36         {
37             MessageBox.Show("Un ou plusieurs champ(s) n'est pas rempli !", ←
38                             "Erreur", MessageBoxButtons.OK, MessageBoxIcon.Error);
39         }
40     }
41 }

```

Listing 47 – ../../Scrum'o'wall/Scrum'o'wall/Views/SprintCreate.xaml.cs

2.45 SprintEdit.xaml

```

1 <!--
2 * Author      :   Ga l Mariot
3 * Project     :   Scrum'o'wall
4 * File        :   SprintEdit.xaml
5 * Desc.       :   This file contains the basic template of the SprintEdit view
6 -->
7 <Window x:Class="Scrum_o_wall.Views.SprintEdit"
8     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
9     xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
10    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
11    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
12    xmlns:local="clr-namespace:Scrum_o_wall.Views"
13    mc:Ignorable="d"
14    Title="Modification d'un sprint" Height="281.121" Width="462.967" ←
15        WindowStartupLocation="CenterScreen"   FontSize="16" >
16     <Grid>
17         <Grid.ColumnDefinitions>
18             <ColumnDefinition Width="140*"/>
19             <ColumnDefinition Width="249*"/>
20         </Grid.ColumnDefinitions>
21         <Grid.RowDefinitions>
22             <RowDefinition Height="84*"/>
23             <RowDefinition Height="43*"/>
24             <RowDefinition Height="43*"/>
25             <RowDefinition Height="53*"/>
26         </Grid.RowDefinitions>
27
28         <Label Grid.ColumnSpan="2" HorizontalAlignment="Center" ←
29             VerticalAlignment="Center" FontSize="36" Margin="10">Modification un ←
30             sprint</Label>
31
32         <Label Grid.Row="1" HorizontalAlignment="Right" ←
33             VerticalAlignment="Center" >Date de d but :</Label>
34         <DatePicker Grid.Row="1" Name="dtpckDateBegin" Grid.Column="1" ←
35             Margin="10"/>
36
37         <Label Grid.Row="2" HorizontalAlignment="Right" ←
38             VerticalAlignment="Center" >Date de fin :</Label>
39         <DatePicker Grid.Row="2" Name="dtpckDateEnd" Grid.Column="1" Margin="10"/>
40
41     <Grid Grid.Row="5" Grid.ColumnSpan="2">

```

```

36         <Grid.ColumnDefinitions>
37             <ColumnDefinition></ColumnDefinition>
38             <ColumnDefinition></ColumnDefinition>
39             <ColumnDefinition Width="2*" /></ColumnDefinition>
40     </Grid.ColumnDefinitions>
41     <Button x:Name="btnCancel" Margin="10" Click="BtnCancel_Click" ↵
42         TouchUp="BtnCancel_Click">Annuler</Button>
43     <Button x:Name="btnDelete" Grid.Column="1" Margin="10" ↵
44         Click="BtnDelete_Click" ↵
45         TouchUp="BtnDelete_Click">Supprimer</Button>
46     <Button x:Name="btnConfirm" Grid.Column="2" Margin="10" ↵
47         Click="BtnConfirm_Click" ↵
48         TouchUp="BtnConfirm_Click">Confirmer</Button>
49     </Grid>
50 </Grid>
51 </Window>

```

Listing 48 – .../Scrum'o'wall/Scrum'o'wall/Views/SprintEdit.xaml

2.46 SprintEdit.xaml.cs

```

1  /*
2   * Author    :    Ga l Serge Mariot
3   * Project   :    Scrum'o'wall
4   * File      :    SprintEdit.xaml.cs
5   * Desc.     :    This file contains the logic in the SprintEdit view
6   */
7  using Scrum_o_wall.Classes;
8  using System;
9  using System.Windows;
10
11 namespace Scrum_o_wall.Views
12 {
13     /// <summary>
14     /// Logique d'interaction pour SprintEdit.xaml
15     /// </summary>
16     public partial class SprintEdit : Window
17     {
18         private readonly Sprint sprint;
19         public bool Deleted = false;
20         public SprintEdit(Sprint aSprint)
21         {
22             sprint = aSprint;
23             InitializeComponent();
24
25             dtpckDateBegin.SelectedDate = sprint.Begin;
26             dtpckDateEnd.SelectedDate = sprint.End;
27         }
28
29         private void BtnCancel_Click(object sender, EventArgs e)
30         {
31
32             Close();
33         }
34         private void BtnConfirm_Click(object sender, EventArgs e)
35         {
36             if (dtpckDateBegin.SelectedDate != null && ↵
37                 dtpckDateEnd.SelectedDate != null)
38             {
39                 DialogResult = true;
40                 Close();
41             }
42             else
43             {
44                 MessageBox.Show("Un ou plusieurs champ(s) n'est pas rempli !", ↵
45                             "Erreur", MessageBoxButtons.OK, MessageBoxIcon.Error);
46             }
47         }
48         private void BtnDelete_Click(object sender, EventArgs e)
49         {
50             if (MessageBox.Show("Le sprint sera supprimé.\nVoulez-vous supprimer ce sprint?", "Attention", MessageBoxButtons.YesNo, MessageBoxIcon.Warning) == MessageBoxResult.Yes)
51             {
52                 DialogResult = true;
53                 Deleted = true;
54                 Close();
55             }
56         }
57     }
58 }

```

```

53         }
54     }
55 }
56 }
57 }

```

Listing 49 – ../../Scrum'o'wall/Scrum'o'wall/Views/SprintEdit.xaml.cs

2.47 SprintMenu.xaml

```

1 <!--
2 * Author      :   Ga l Mariot
3 * Project     :   Scrum'o'wall
4 * File        :   SprintMenu.xaml
5 * Desc.       :   This file contains the basic template of the SprintMenu view
6 -->
7 <Window x:Class="Scrum_o_wall.Views.SprintMenu"
8     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
9     xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
10    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
11    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
12    xmlns:local="clr-namespace:Scrum_o_wall.Views"
13    mc:Ignorable="d"
14    Title="SprintMenu" Height="450" Width="800" WindowState="Maximized" -->
15    WindowStyle="None" WindowStartupLocation="CenterScreen" -->
16    ResizeMode="NoResize" FontSize="16">
17 <Window.ContextMenu >
18     <ContextMenu >
19         <MenuItem Header="Ajouter une colonne" Name="addColumn" -->
20             Click="AddColumn_Click"/>
21         <MenuItem Header="Ouvrir le burndown chart" -->
22             Name="btnBurndownChartMenu" Click="BtnBurndownChart_Click"/>
23         <MenuItem Header="Editer le sprint" Name="btnEditSprint" -->
24             Click="BtnEditSprint_Click"/>
25         <Separator/>
26         <MenuItem Header="Retour" Name="Quit" Click="BtnReturn_Click"/>
27     </ContextMenu>
28 </Window.ContextMenu>
29 <Canvas x:Name="cnvsSprint">
30     <Label Name="lblProjectName" Content="Nom du projet" FontSize="30" -->
31         Height="60"/>
32     <Label Name="lblSprintName" Content="Nom du sprint" Canvas.Top="65" -->
33         Height="30"/>
34     <Button TouchUp="BtnReturn_Click" x:Name="btnReturn" -->
35         Click="BtnReturn_Click" Content=" " -->
36         VerticalContentAlignment="Top" Canvas.Bottom="10" Width="75" -->
37         Height="75" BorderBrush="Black" FontSize="45" Canvas.Left="80">
38         <Button.Resources>
39             <Style TargetType="Border">
40                 <Setter Property="CornerRadius" Value="50"/>
41             </Style>
42         </Button.Resources>
43     </Button>
44     <Button TouchUp="BtnBurndownChart_Click" x:Name="btnBurndownChart" -->
45         Click="BtnBurndownChart_Click" Content=" " -->
46         VerticalContentAlignment="Top" Canvas.Bottom="10" Width="75" -->
47         Height="75" BorderBrush="Black" FontSize="45">
48         <Button.Resources>
49             <Style TargetType="Border">
50                 <Setter Property="CornerRadius" Value="50"/>
51             </Style>
52         </Button.Resources>
53     </Button>
54 </Canvas>
55 </Window>

```

Listing 50 – ../../Scrum'o'wall/Scrum'o'wall/Views/SprintMenu.xaml

2.48 SprintMenu.xaml.cs

```

1 /*
2 * Author      :   Ga l Serge Mariot
3 * Project     :   Scrum'o'wall
4 * File        :   SprintMenu.xaml.cs

```

```

5 * Desc.      : This file contains the logic in the SprintMenu view
6 */
7 using Scrum_o_wall.Classes;
8 using System;
9 using System.Collections.Generic;
10 using System.Linq;
11 using System.Windows;
12 using System.Windows.Controls;
13 using System.Windows.Input;
14 using System.Windows.Media;
15
16 namespace Scrum_o_wall.Views
17 {
18     /// <summary>
19     /// Logique d'interaction pour SprintMenu.xaml
20     /// </summary>
21     public partial class SprintMenu : Window
22     {
23         private readonly Sprint sprint;
24         private readonly Controller controller;
25         private readonly List<GroupBox> columns = new List<GroupBox>();
26         private readonly List<UserControl> userStories = new List<UserControl>();
27
28         private readonly Dictionary<InputDevice, Point> currentPoint = new ←
29             Dictionary<InputDevice, Point>();
30         private readonly Dictionary<InputDevice, UserControl> infos = new ←
31             Dictionary<InputDevice, UserControl>();
32
33         public SprintMenu(Sprint aSprint, Controller aController)
34         {
35             InitializeComponent();
36             sprint = aSprint;
37             controller = aController;
38
39             Loaded += SprintMenu_Loaded;
40             PreviewTouchMove += SprintMenu_PreviewTouchMove;
41             TouchUp += SprintMenu_PreviewTouchUp;
42
43             lblProjectName.Content = sprint.Project.Name;
44             lblSprintName.Content = aSprint.ToString();
45
46             CleanLists();
47         }
48
49         private void CleanLists()
50         {
51             foreach (GroupBox gbx in columns)
52             {
53                 cnvsSprint.Children.Remove(gbx);
54             }
55             columns.Clear();
56             foreach (UserControl userControl in userStories)
57             {
58                 cnvsSprint.Children.Remove(userControl);
59             }
59             userStories.Clear();
60
61             /// <summary>
62             /// Refresh the view with its content
63             /// </summary>
64             private void Refresh()
65             {
66                 CleanLists();
67
68                 // Declare variables for userstories positioning
69                 Dictionary<State, int> userStoriesPerState = new Dictionary<State, ←
70                     int>();
71
72                 foreach (KeyValuePair<int, State> keyValuePair in ←
73                     sprint.Project.States)
74                 {
75                     State state = keyValuePair.Value;
76                     CreateStateColumn(state);
77                     userStoriesPerState.Add(state, 0);
78                 }
79
80                 /// Place UserStories
81                 foreach (KeyValuePair<int, UserStory> item in ←
82                     sprint.OrderedUserStories)
83                 {

```

```

79         UserStory userStory = item.Value;
80         CreateUserStoryControl(userStory, ←
81             userStoriesPerState[userStory.State]);
82         userStoriesPerState[userStory.State]++;
83     }
84     private GroupBox CreateStateColumn(State state)
85     {
86         //Create the column
87         GroupBox gbx = new GroupBox
88     {
89         Height = cnvsSprint.ActualHeight - 190,
90         Width = cnvsSprint.ActualWidth / sprint.Project.States.Count,
91         Name = "gbx" + state.Name.Replace(" ", "").Replace("'", ""),
92         Header = state.Name,
93         Tag = state,
94         BorderBrush = Brushes.Black,
95         BorderThickness = new Thickness(1)
96     };
97
98         //Positioning and put into canvas
99         cnvsSprint.Children.Add(gbx);
100        Canvas.SetTop(gbx, 100);
101        Canvas.SetBottom(gbx, 90);
102        Canvas.SetLeft(gbx, gbx.Width * columns.Count);
103
104        //added to list and returned
105        columns.Add(gbx);
106        return gbx;
107    }
108
109
110    private UserControl CreateUserStoryControl(UserStory userStory, int ←
111        cptTop = 0)
112    {
113        GroupBox gbx = columns.Where(c => c.Tag == userStory.State).First();
114
115        //Create userStory frame
116        TextBlock content = new TextBlock
117        {
118            Text = userStory.ToString(),
119            TextWrapping = TextWrapping.Wrap
120        };
121
122        UserControl userControl = new UserControl
123        {
124            Content = content,
125            Tag = userStory,
126            Height = 50,
127            Width = gbx.Width - 20,
128            BorderBrush = Brushes.Black,
129            Cursor = Cursors.Hand
130        };
131        // Set background color by limit date
132        if (userStory.DateLimit != null)
133        {
134            if (DateTime.Now > userStory.DateLimit)
135            {
136                userControl.Background = Brushes.LightBlue;
137            }
138            else
139            {
140                userControl.Background = Brushes.LightPink;
141            }
142        }
143        else
144        {
145            userControl.Background = Brushes.LightGray;
146        }
147        userControl.MouseDoubleClick += UsrCtrlUserStory_Click;
148        userControl.TouchUp += UsrCtrlUserStory_Click;
149
150        //Events for drag'n'drop
151        userControl.PreviewMouseDown += UserStory_PreviewMouseDown;
152        Stylus.SetIsPressAndHoldEnabled(userControl, false);
153
154        //Add to lists, positionning and return
155        cnvsSprint.Children.Add(userControl);
156        Canvas.SetLeft(userControl, Canvas.GetLeft(gbx) + 10);

```

```

156         Canvas.SetTop(userControl, Canvas.GetTop(gbx) + 30 + 60 * cptTop);
157         userStories.Add(userControl);
158     }
159 }
160
161 private void SprintMenu_Loaded(object sender, RoutedEventArgs e)
162 {
163     cnvsSprint.Width = ActualWidth;
164     cnvsSprint.Height = ActualHeight;
165
166     //set control positions
167     Canvas.SetLeft(lblProjectName, (cnvsSprint.Width - ←
168         lblProjectName.ActualWidth) / 2.0);
169     Canvas.SetLeft(lblSprintName, (cnvsSprint.Width - ←
170         lblSprintName.ActualWidth) / 2.0);
171     Canvas.SetLeft(btnBurndownChart, (cnvsSprint.Width) / 2.0 + ←
172         btnBurndownChart.ActualWidth);
173     Canvas.SetLeft(btnReturn, (cnvsSprint.Width) / 2.0 - ←
174         btnReturn.ActualWidth);
175
176     Refresh();
177 }
178
179 private void SprintMenu_PreviewTouchUp(object sender, TouchEventArgs e)
180 {
181     if (currentPoint.ContainsKey(e.Device))
182     {
183         //Search the contained groupbox
184         Point p = e.GetTouchPoint(null).Position;
185         GroupBox gbxState = null;
186         foreach (GroupBox col in columns)
187         {
188             double leftBound = Canvas.GetLeft(col);
189             double rightBound = leftBound + col.Width;
190             double topBound = Canvas.GetTop(col);
191             double bottomBound = topBound + col.Height;
192             if (p.X > leftBound && p.X < rightBound &&
193                 p.Y > topBound && p.Y < bottomBound)
194             {
195                 gbxState = col;
196                 break;
197             }
198         }
199         //if release in a groupbox, change state
200         if (gbxState != null)
201         {
202             gbxState.BorderThickness = new Thickness(1);
203
204             State state = gbxState.Tag as State;
205             UserStory userStory = (infos[e.Device] as UserControl).Tag ←
206                 as UserStory;
207             controller.UserStorySwitchState(userStory, state);
208         }
209         currentPoint.Remove(e.Device);
210         infos.Remove(e.Device);
211         Refresh();
212     }
213 }
214
215 private void SprintMenu_PreviewTouchMove(object sender, TouchEventArgs e)
216 {
217     //Update position of point
218     if (currentPoint.ContainsKey(e.Device))
219     {
220         currentPoint[e.Device] = e.GetTouchPoint(null).Position;
221     }
222     //Set border thickness to default
223     foreach (GroupBox col in columns)
224     {
225         col.BorderThickness = new Thickness(1);
226     }
227     //Verify each points to change border thickness if in bounds
228     foreach (KeyValuePair<InputDevice, Point> keyValuePair in ←
229         currentPoint)
230     {
231         Point p = keyValuePair.Value;
232         foreach (GroupBox col in columns)
233         {
234             double leftBound = Canvas.GetLeft(col);
235             double rightBound = leftBound + col.Width;
236             double topBound = Canvas.GetTop(col);

```

```

229         double bottomBound = topBound + col.Height;
230
231         if (p.X > leftBound && p.X < rightBound &&
232             p.Y > topBound && p.Y < bottomBound)
233         {
234             col.BorderThickness = new Thickness(3);
235         }
236     }
237 }
238
239 private void UserStory_PreviewTouchDown(object sender, TouchEventArgs e)
240 {
241     if (currentPoint.ContainsKey(e.Device))
242     {
243         currentPoint.Remove(e.Device);
244         infos.Remove(e.Device);
245     }
246
247     currentPoint.Add(e.Device, e.GetTouchPoint(null).Position);
248     infos.Add(e.Device, sender as UserControl);
249 }
250
251 private void State_DragLeave(object sender, DragEventArgs e)
252 {
253     GroupBox gbx = sender as GroupBox;
254     gbx.BorderThickness = new Thickness(1);
255 }
256
257 private void State_DragEnter(object sender, DragEventArgs e)
258 {
259     GroupBox gbx = sender as GroupBox;
260     gbx.BorderThickness = new Thickness(5);
261 }
262
263 private void State_Drop(object sender, DragEventArgs e)
264 {
265     GroupBox gbx = sender as GroupBox;
266     gbx.BorderThickness = new Thickness(1);
267
268     //Make userStory switch
269     State state = gbx.Tag as State;
270     UserStory userStory = e.Data.GetData("drag") as UserStory;
271     controller.UserStorySwitchState(userStory, state);
272
273     Refresh();
274 }
275
276 private void UsrCtrlUserStory_Click(object sender, EventArgs e)
277 {
278     UserStory userStory = (sender as UserControl).Tag as UserStory;
279     UserStoryEdit userStoryEdit = new UserStoryEdit(userStory, ←
280         sprint.Project, controller);
281     if (userStoryEdit.ShowDialog() == true)
282     {
283         if (userStoryEdit.Deleted)
284         {
285             controller.Delete(userStory);
286         }
287         else
288         {
289             string desc = userStoryEdit.tbxDesc.Text.Trim();
290             DateTime? dateLimite = ←
291                 userStoryEdit.dtpckrDateLimit.SelectedDate;
292             int complexity = ←
293                 Convert.ToInt32(userStoryEdit.tbxComplexity.Text);
294             int completedComplexity = ←
295                 Convert.ToInt32(userStoryEdit.tbxCompletedComplexity.Text);
296             bool blocked = userStoryEdit.chkBxBlocked.IsChecked == true;
297             Priority priority = ←
298                 (Priority)userStoryEdit.cbxPriority.SelectedItem;
299             Classes.Type type = ←
300                 (Classes.Type)userStoryEdit.cbxType.SelectedItem;
301             State state = userStory.State;
302
303             controller.UpdateUserStory(desc, dateLimite, complexity, ←
304                 completedComplexity, blocked, priority, type, state, ←
305                 userStory);
306         }
307         Refresh();
308     }
309 }
310
311 private void AddColumn_Click(object sender, RoutedEventArgs e)
312 {

```

```

300         StateMenu stateMenu = new StateMenu(sprint.Project, controller);
301         stateMenu.ShowDialog();
302         Refresh();
303     }
304     private void BtnBurndownChart_Click(object sender, EventArgs e)
305     {
306         BurndownChart burndownChart = new BurndownChart(sprint);
307         burndownChart.ShowDialog();
308     }
309     private void BtnReturn_Click(object sender, EventArgs e)
310     {
311         Close();
312     }
313
314     private void BtnEditSprint_Click(object sender, RoutedEventArgs e)
315     {
316         SprintEdit sprintEdit = new SprintEdit(sprint);
317         if (sprintEdit.ShowDialog() == true)
318         {
319             if (sprintEdit.Deleted)
320             {
321                 controller.Delete(sprint);
322                 DialogResult = true;
323                 Close();
324             }
325             else
326             {
327                 DateTime begin = sprintEdit.dtpckDateBegin.SelectedDate == null ? DateTime.Now : sprintEdit.dtpckDateBegin.SelectedDate.Value;
328                 DateTime end = sprintEdit.dtpckDateEnd.SelectedDate == null ? DateTime.Now + new TimeSpan(7, 0, 0) : sprintEdit.dtpckDateEnd.SelectedDate.Value;
329                 controller.UpdateSprint(begin, end, sprint);
330                 Refresh();
331             }
332         }
333     }
334 }
335

```

Listing 51 – ../../Scrum'o'wall/Scrum'o'wall/Views/SprintMenu.xaml.cs

2.49 StateCreate.xaml

```

1 <!--
2 * Author    :   Gael Mariot
3 * Project   :   Scrum'o'wall
4 * File      :   StateCreate.xaml
5 * Desc.     :   This file contains the basic template of the StateCreate view
6 -->
7 <Window x:Class="Scrum_o_wall.Views.StateCreate"
8     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
9     xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
10    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
11    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
12    xmlns:local="clr-namespace:Scrum_o_wall.Views"
13    mc:Ignorable="d"
14    Title="Cr ation d'un tat " Height="169.532" Width="500.533" -->
15    WindowStartupLocation="CenterScreen" FontSize="16">
16    <Grid>
17        <Grid.ColumnDefinitions>
18            <ColumnDefinition Width="174" /></ColumnDefinition>
19            <ColumnDefinition /></ColumnDefinition>
20        </Grid.ColumnDefinitions>
21        <Grid.RowDefinitions>
22            <RowDefinition /></RowDefinition>
23            <RowDefinition /></RowDefinition>
24            <RowDefinition /></RowDefinition>
25        </Grid.RowDefinitions>
26        <TextBlock Grid.ColumnSpan="2" TextAlignment="Center" FontSize="20" -->
27            VerticalAlignment="Center">Cr ation d'un tat </TextBlock>
28        <TextBlock Grid.Row="1" TextAlignment="Right" -->
29            VerticalAlignment="Center" Margin="10,13,10,14">Nom de l' tat   <!--
30            :</TextBlock>
31        <TextBox Grid.Row="1" Grid.Column="1" MaxLength="30" Margin="10" -->
32            Name="tbxStateName" Stylus.IsPressAndHoldEnabled="False"></TextBox>

```

```

28         <Button TouchUp="BtnCancel_Click" x:Name="btnCancel" Grid.Row="2" ←
29             Grid.Column="0" Margin="10" Click="BtnCancel_Click">Annuler</Button>
30         <Button TouchUp="BtnConfirm_Click" x:Name="btnConfirm" Grid.Row="2" ←
31             Grid.Column="1" Margin="10" Click="BtnConfirm_Click">Confirmer</Button>
32     </Grid>
33 </Window>

```

Listing 52 – ../../Scrum'o'wall/Scrum'o'wall/Views/StateCreate.xaml

2.50 StateCreate.xaml.cs

```

1 /*
2 * Author : Ga l Serge Mariot
3 * Project : Scrum'o'wall
4 * File : StateCreate.xaml.cs
5 * Desc. : This file contains the logic in the StateCreate view
6 */
7 using System;
8 using System.Windows;
9
10 namespace Scrum_o_wall.Views
11 {
12     /// <summary>
13     /// Logique d'interaction pour StateCreate.xaml
14     /// </summary>
15     public partial class StateCreate : Window
16     {
17         public StateCreate()
18         {
19             InitializeComponent();
20         }
21
22         private void BtnCancel_Click(object sender, EventArgs e)
23         {
24
25             Close();
26         }
27         private void BtnConfirm_Click(object sender, EventArgs e)
28         {
29             if (tbxStateName.Text.Trim().Length > 0)
30             {
31                 DialogResult = true;
32                 Close();
33             }
34             else
35             {
36                 MessageBox.Show("Un ou plusieurs champ(s) n'est pas rempli !", ←
37                     "Erreur", MessageBoxButtons.OK, MessageBoxIcon.Error);
38             }
39         }
40     }
41 }

```

Listing 53 – ../../Scrum'o'wall/Scrum'o'wall/Views/StateCreate.xaml.cs

2.51 StateEdit.xaml

```

1 <!--
2 * Author : Ga l Mariot
3 * Project : Scrum'o'wall
4 * File : StateEdit.xaml
5 * Desc. : This file contains the basic template of the StateEdit view
6 -->
7 <Window x:Class="Scrum_o_wall.Views.StateEdit"
8     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
9     xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
10    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
11    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
12    xmlns:local="clr-namespace:Scrum_o_wall.Views"
13    mc:Ignorable="d"
14        Title="Modification d'un tat " Height="169.532" Width="500.533" ←
15            WindowStartupLocation="CenterScreen" FontSize="16">
16     <Grid>

```

```

16     <Grid.ColumnDefinitions>
17         <ColumnDefinition Width="174"></ColumnDefinition>
18         <ColumnDefinition></ColumnDefinition>
19     </Grid.ColumnDefinitions>
20     <Grid.RowDefinitions>
21         <RowDefinition></RowDefinition>
22         <RowDefinition></RowDefinition>
23         <RowDefinition></RowDefinition>
24     </Grid.RowDefinitions>
25     <TextBlock Grid.ColumnSpan="2" TextAlignment="Center" FontSize="20" ↵
26         VerticalAlignment="Center">Modification d'un tat </TextBlock>
27     <TextBlock Grid.Row="1" TextAlignment="Right" ↵
28         VerticalAlignment="Center" Margin="10,13,10,14">Nom de l' tat <-
29         :</TextBlock>
30     <TextBox Grid.Row="1" Grid.Column="1" MaxLength="30" Margin="10" ↵
31         Name="tbxStateName" Stylus.IsPressAndHoldEnabled="False"></TextBox>
32     <Grid Grid.Row="5" Grid.ColumnSpan="2">
33         <Grid.ColumnDefinitions>
34             <ColumnDefinition></ColumnDefinition>
35             <ColumnDefinition></ColumnDefinition>
36             <ColumnDefinition Width="2*"></ColumnDefinition>
37         </Grid.ColumnDefinitions>
38         <Button x:Name="btnCancel" Margin="10" Click="BtnCancel_Click" ↵
39             TouchUp="BtnCancel_Click">Annuler</Button>
40         <Button x:Name="btnDelete" Grid.Column="1" Margin="10" ↵
41             Click="BtnDelete_Click" ↵
42             TouchUp="BtnDelete_Click">Supprimer</Button>
43         <Button x:Name="btnConfirm" Grid.Column="2" Margin="10" ↵
44             Click="BtnConfirm_Click" ↵
45             TouchUp="BtnConfirm_Click">Confirmer</Button>
46     </Grid>
47 </Grid>
48 </Window>

```

Listing 54 – ../../Scrum'o'wall/Scrum'o'wall/Views/StateEdit.xaml

2.52 StateEdit.xaml.cs

```

1 /*
2 * Author   :   Ga l Serge Mariot
3 * Project  :   Scrum'o'wall
4 * File     :   StateEdit.xaml.cs
5 * Desc.    :   This file contains the logic in the StateEdit view
6 */
7 using Scrum_o_wall.Classes;
8 using System;
9 using System.Windows;
10
11 namespace Scrum_o_wall.Views
12 {
13     /// <summary>
14     /// Logique d'interaction pour StateEdit.xaml
15     /// </summary>
16     public partial class StateEdit : Window
17     {
18         private readonly State state;
19         public bool Deleted = false;
20         public StateEdit(State aState)
21         {
22             state = aState;
23             InitializeComponent();
24
25             tbxStateName.Text = state.Name;
26         }
27
28         private void BtnCancel_Click(object sender, EventArgs e)
29         {
30
31             Close();
32         }
33         private void BtnConfirm_Click(object sender, EventArgs e)
34         {
35             if (tbxStateName.Text.Trim().Length > 0)
36             {
37                 DialogResult = true;
38                 Close();
39             }

```

```

40         else
41     {
42         MessageBox.Show("Un ou plusieurs champ(s) n'est pas rempli!", ←
43             "Erreur", MessageBoxButtons.OK, MessageBoxIcon.Error);
44     }
45     private void BtnDelete_Click(object sender, EventArgs e)
46     {
47         if (MessageBox.Show("L'état sera supprimé.\nêtes-vous sûr(e)?", ←
48             "Attention", MessageBoxButtons.YesNo, MessageBoxIcon.Warning) == ←
49             DialogResult.Yes)
50         {
51             DialogResult = true;
52             Deleted = true;
53             Close();
54         }
55     }

```

Listing 55 – ../../Scrum'o'wall/Scrum'o'wall/Views/StateEdit.xaml.cs

2.53 StateMenu.xaml

```

1 <!--
2 * Author      :   Gael Mariot
3 * Project     :   Scrum'o'wall
4 * File        :   StateMenu.xaml
5 * Desc.       :   This file contains the basic template of the StateMenu view
6 -->
7 <Window x:Class="Scrum_o_wall.Views.StateMenu"
8     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
9     xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
10    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
11    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
12    xmlns:local="clr-namespace:Scrum_o_wall.Views"
13    mc:Ignorable="d"
14    Title="Gestion des états" Height="450" Width="600" ←
15        Closing="StateMenu_Closing" WindowStartupLocation="CenterScreen" ←
16        FontSize="16" ResizeMode="NoResize">
17     <Canvas x:Name="cnvsStates" Margin="0">
18         <Grid x:Name="grdStates" Width="594" Height="421">
19             <Grid.ColumnDefinitions>
20                 <ColumnDefinition Width="80*"/></ColumnDefinition>
21                 <ColumnDefinition Width="20*"/></ColumnDefinition>
22                 <ColumnDefinition Width="80*"/></ColumnDefinition>
23             </Grid.ColumnDefinitions>
24             <Grid.RowDefinitions>
25                 <RowDefinition Height="7*"/></RowDefinition>
26                 <RowDefinition Height="40*"/></RowDefinition>
27             </Grid.RowDefinitions>
28             <TextBlock Grid.ColumnSpan="3" FontSize="32" TextAlignment="Center" ←
29                 VerticalAlignment="Center" >Gestion des états </TextBlock>
30             <GroupBox Margin="10" Header="États possibles" Grid.Row="1">
31                 <Grid>
32                     <Grid.RowDefinitions>
33                         <RowDefinition/></RowDefinition>
34                         <RowDefinition Height="70"/></RowDefinition>
35                     </Grid.RowDefinitions>
36                     <ListBox Margin="10" Name="lstPossibleStates" ←
37                         MouseDoubleClick="Lst_MouseDoubleClick" ←
38                         Stylus.IsPressAndHoldEnabled="False">
39                         </ListBox>
40                         <Button TouchUp="BtnAddState_Click" ←
41                             Click="BtnAddState_Click" x:Name="btnAddState" ←
42                             Margin="10" Grid.Row="1">Ajouter</Button>
43                     </Grid>
44                 </GroupBox>
45                 <GroupBox Grid.Column="2" Margin="10" Header="États assignés" ←
46                     Grid.Row="1">
47                     <ListBox Margin="10" Name="lstAssignedStates" ←
48                         Stylus.IsPressAndHoldEnabled="False">
49                         </ListBox>
50                 </GroupBox>

```

```

44     <Grid Grid.Row="1" Grid.Column="1">
45         <Grid.RowDefinitions>
46             <RowDefinition/></RowDefinition>
47             <RowDefinition/></RowDefinition>
48             <RowDefinition/></RowDefinition>
49     </Grid.RowDefinitions>
50     <Button TouchUp="BtnGoLeft_Click" Click="BtnGoLeft_Click" ←
51         Margin="10" x:Name="btnGoLeft" Content="à " ></Button>
52     <Button TouchUp="BtnGoRight_Click" Click="BtnGoRight_Click" ←
53         Margin="10" x:Name="btnGoRight" Grid.Row="1" ←
54         Content="â " ></Button>
55     <Button TouchUp="BtnSave_Click" Click="BtnSave_Click" ←
56         Margin="10,10,10,29" x:Name="btnSave" Grid.Row="2" ←
57         Content="    " ></Button>
58     </Grid>
59   </Grid>
60 </Canvas>
61 </Window>

```

Listing 56 – ../../Scrum'o'wall/Scrum'o'wall/Views/StateMenu.xaml

2.54 StateMenu.xaml.cs

```

1  /*
2  * Author      : Gaël Serge Mariot
3  * Project     : Scrum'o'wall
4  * File        : StateMenu.xaml.cs
5  * Desc.       : This file contains the logic in the StateMenu view
6  */
7  using Scrum_o_wall.Classes;
8  using System;
9  using System.Collections.Generic;
10 using System.Linq;
11 using System.Windows;
12 using System.Windows.Controls;
13 using System.Windows.Input;
14
15 namespace Scrum_o_wall.Views
16 {
17     /// <summary>
18     /// Logique d'interaction pour StateMenu.xaml
19     /// </summary>
20     public partial class StateMenu : Window
21     {
22         private readonly Project project;
23         private readonly Controller controller;
24         public StateMenu(Project aProject, Controller aController)
25         {
26             project = aProject;
27             controller = aController;
28
29             InitializeComponent();
30
31             Refresh();
32         }
33
34         private void Refresh()
35         {
36             if (lstAssignedStates.Items.Count == 0)
37             {
38                 foreach (KeyValuePair<int, State> keyValuePair in project.States)
39                 {
40                     lstAssignedStates.Items.Add(keyValuePair.Value);
41                 }
42             }
43             lstPossibleStates.Items.Clear();
44             foreach (State state in controller.States.Where(s => ←
45                 !lstAssignedStates.Items.Contains(s)))
46             {
47                 lstPossibleStates.Items.Add(state);
48             }
49         }
50
51         private void StateMenu_Closing(object sender, EventArgs e)
52         {
53             if (MessageBox.Show("Les changements non sauvegardés seront perdus.\nVoulez-vous sauvegarder les modifications?", ←

```

```

        "Attention", MessageBoxButtons.YesNo, MessageBoxIcon.Warning) == DialogResult.Yes)
53    {
54        Save();
55    }
56    Close();
57}
58private void BtnGoLeft_Click(object sender, EventArgs e)
59{
60    if (lstAssignedStates.SelectedItem is State state)
61    {
62        lstAssignedStates.Items.Remove(state);
63        lstPossibleStates.Items.Add(state);
64    }
65}
66private void BtnGoRight_Click(object sender, EventArgs e)
67{
68    if (lstPossibleStates.SelectedItem is State state)
69    {
70        lstPossibleStates.Items.Remove(state);
71        lstAssignedStates.Items.Add(state);
72    }
73}
74private void Save()
75{
76    List<State> toRemove = new List<State>();
77    List<State> toAdd = new List<State>();
78    foreach (State state in controller.States)
79    {
80        if (lstAssignedStates.Items.Contains(state) && !project.States.ContainsValue(state))
81        {
82            toAdd.Add(state);
83        }
84        else if (lstPossibleStates.Items.Contains(state) && project.States.ContainsValue(state))
85        {
86            toRemove.Add(state);
87        }
88    }
89    foreach (State state in toAdd)
90    {
91        controller.AddStateToProject(state, project);
92    }
93    foreach (State state in toRemove)
94    {
95        controller.RemoveStateFromProject(state, project);
96    }
97}
98private void BtnSave_Click(object sender, EventArgs e)
99{
100    Save();
101    Close();
102}
103private void BtnAddState_Click(object sender, EventArgs e)
104{
105    StateCreate stateCreate = new StateCreate();
106    if (stateCreate.ShowDialog() == true)
107    {
108        controller.CreateState(stateCreate.tbxStateName.Text);
109        Refresh();
110    }
111}
112
113private void Lst_MouseDoubleClick(object sender, MouseButtonEventArgs e)
114{
115    State state = (sender as ListBox).SelectedItem as State;
116    StateEdit stateEdit = new StateEdit(state);
117    if (stateEdit.ShowDialog() == true)
118    {
119        if (stateEdit.Deleted)
120        {
121            controller.Delete(state);
122        }
123        else
124        {
125            string name = stateEdit.tbxStateName.Text.Trim();
126            controller.UpdateState(name, state);
127        }

```

```

128         Refresh();
129     }
130 }
131 }
132 }
```

Listing 57 – ../../Scrum'o'wall/Scrum'o'wall/Views/StateMenu.xaml.cs

2.55 UserCreate.xaml

```

1 <!--
2 * Author      :   Ga l Mariot
3 * Project     :   Scrum'o'wall
4 * File        :   UserCreate.xaml
5 * Desc.       :   This file contains the basic template of the UserCreate view
6 -->
7 <Window x:Class="Scrum_o_wall.Views.UserCreate"
8     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
9     xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
10    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
11    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
12    xmlns:local="clr-namespace:Scrum_o_wall.Views"
13    mc:Ignorable="d"
14    Title="Cr ation d'un utilisateur" Height="178.199" Width="458.533" ↵
15        WindowStartupLocation="CenterScreen" FontSize="16">
16     <Grid>
17         <Grid.ColumnDefinitions>
18             <ColumnDefinition Width="179" /></ColumnDefinition>
19             <ColumnDefinition /></ColumnDefinition>
20         </Grid.ColumnDefinitions>
21         <Grid.RowDefinitions>
22             <RowDefinition /></RowDefinition>
23             <RowDefinition /></RowDefinition>
24             <RowDefinition /></RowDefinition>
25         </Grid.RowDefinitions>
26         <TextBlock Grid.ColumnSpan="2" TextAlignment="Center" FontSize="20" ↵
27             VerticalAlignment="Center">Cr ation d'un utilisateur</TextBlock>
28         <TextBlock Grid.Row="1" TextAlignment="Right" ↵
29             VerticalAlignment="Center" Margin="10,13,10,14">Nom de l'utilisateur ↵
30             :</TextBlock>
31         <TextBox Grid.Row="1" Name="tbxUserName" Grid.Column="1" ↵
32             MaxLength="255" Margin="10" ↵
33             Stylus.IsPressAndHoldEnabled="False"></TextBox>
34         <Button TouchUp="BtnCancel_Click" x:Name="btnCancel" ↵
35             Click="BtnCancel_Click" Grid.Row="2" Grid.Column="0" ↵
36             Margin="10">Annuler</Button>
37         <Button TouchUp="BtnConfirm_Click" x:Name="btnConfirm" ↵
38             Click="BtnConfirm_Click" Grid.Row="2" Grid.Column="1" ↵
39             Margin="10">Confirmer</Button>
40     </Grid>
41 </Window>
```

Listing 58 – ../../Scrum'o'wall/Scrum'o'wall/Views/UserCreate.xaml

2.56 UserCreate.xaml.cs

```

1 /*
2 * Author      :   Ga l Serge Mariot
3 * Project     :   Scrum'o'wall
4 * File        :   UserCreate.xaml.cs
5 * Desc.       :   This file contains the logic in the UserCreate view
6 */
7 using System;
8 using System.Windows;
9
10 namespace Scrum_o_wall.Views
11 {
12     /// <summary>
13     /// Logique d'interaction pour UserCreate.xaml
14     /// </summary>
15     public partial class UserCreate : Window
16     {
17         public UserCreate()
18         {
```

```

19         InitializeComponent();
20     }
21
22     private void btnCancel_Click(object sender, EventArgs e)
23     {
24
25         Close();
26     }
27     private void btnConfirm_Click(object sender, EventArgs e)
28     {
29         if (txbUserName.Text.Trim().Length > 0)
30         {
31             DialogResult = true;
32             Close();
33         }
34     }
35
36 }
37 }
```

Listing 59 – ../../Scrum'o'wall/Scrum'o'wall/Views/UserCreate.xaml.cs

2.57 UserEdit.xaml

```

1 <!--
2 * Author      :   Ga l Mariot
3 * Project     :   Scrum'o'wall
4 * File        :   UserEdit.xaml
5 * Desc.       :   This file contains the basic template of the UserEdit view
6 -->
7 <Window x:Class="Scrum_o_wall.Views.UserEdit"
8     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
9     xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
10    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
11    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
12    xmlns:local="clr-namespace:Scrum_o_wall.Views"
13    mc:Ignorable="d"
14    Title="Modification d'un utilisateur" Height="166.987" Width="485.083" ←
15        WindowStartupLocation="CenterScreen" FontSize="16">
16     <Grid>
17         <Grid.ColumnDefinitions>
18             <ColumnDefinition Width="173" /></ColumnDefinition>
19             <ColumnDefinition /></ColumnDefinition>
20         </Grid.ColumnDefinitions>
21         <Grid.RowDefinitions>
22             <RowDefinition /></RowDefinition>
23             <RowDefinition /></RowDefinition>
24             <RowDefinition /></RowDefinition>
25         </Grid.RowDefinitions>
26         <TextBlock Grid.ColumnSpan="2" TextAlignment="Center" FontSize="20" ←
27             VerticalAlignment="Center">Modification d'un utilisateur</TextBlock>
28         <TextBlock Grid.Row="1" TextAlignment="Right" ←
29             VerticalAlignment="Center" Margin="10,13,10,14">Nom de l'utilisateur ←
30             :</TextBlock>
31         <TextBox Grid.Row="1" Name="txbUserName" Grid.Column="1" ←
32             MaxLength="255" Margin="10" ←
33             Stylus.IsPressAndHoldEnabled="False" /></TextBox>
34         <Grid Grid.Row="5" Grid.ColumnSpan="2">
35             <Grid.ColumnDefinitions>
36                 <ColumnDefinition /></ColumnDefinition>
37                 <ColumnDefinition /></ColumnDefinition>
38                 <ColumnDefinition Width="2*" /></ColumnDefinition>
39             </Grid.ColumnDefinitions>
40             <Button x:Name="btnCancel" Margin="10" Click="BtnCancel_Click" ←
41                 TouchUp="BtnCancel_Click">Annuler</Button>
42             <Button x:Name="btnDelete" Grid.Column="1" Margin="10" ←
43                 Click="BtnDelete_Click" ←
44                 TouchUp="BtnDelete_Click">Supprimer</Button>
45             <Button x:Name="btnConfirm" Grid.Column="2" Margin="10" ←
46                 Click="BtnConfirm_Click" ←
47                 TouchUp="BtnConfirm_Click">Confirmer</Button>
48         </Grid>
49     </Grid>
50 </Window>
```

Listing 60 – ../../Scrum'o'wall/Scrum'o'wall/Views/UserEdit.xaml

2.58 UserEdit.xaml.cs

```
1  /*
2   * Author    :    Ga l Serge Mariot
3   * Project   :    Scrum'o'wall
4   * File      :    UserEdit.xaml.cs
5   * Desc.     :    This file contains the logic in the UserEdit view
6   */
7  using Scrum_o_wall.Classes;
8  using System;
9  using System.Windows;
10
11 namespace Scrum_o_wall.Views
12 {
13     /// <summary>
14     /// Logique d'interaction pour UserEdit.xaml
15     /// </summary>
16     public partial class UserEdit : Window
17     {
18         private readonly User user;
19         public bool Deleted = false;
20         public UserEdit(User aUser)
21         {
22             user = aUser;
23
24             InitializeComponent();
25
26             tbxUserName.Text = user.Name;
27         }
28
29         private void BtnConfirm_Click(object sender, EventArgs e)
30         {
31             if (tbxUserName.Text.Trim().Length > 0)
32             {
33                 DialogResult = true;
34                 Close();
35             }
36             else
37             {
38                 MessageBox.Show("Un ou plusieurs champ(s) n'est pas rempli !", ←
39                               "Erreur", MessageBoxButtons.OK, MessageBoxIcon.Error);
39             }
40         }
41         private void btnCancel_Click(object sender, EventArgs e)
42         {
43
44             Close();
45         }
46         private void BtnDelete_Click(object sender, EventArgs e)
47         {
48             if (MessageBox.Show("L'utilisateur sera supprim .\n tes -vous ←
49                           s r(e)?", "Attention", MessageBoxButtons.YesNo, ←
50                           MessageBoxIcon.Warning) == MessageBoxResult.Yes)
51             {
52                 Deleted = true;
53                 DialogResult = true;
54                 Close();
55             }
56         }
57     }
}
```

Listing 61 – ../../Scrum'o'wall/Scrum'o'wall/Views/UserEdit.xaml.cs

2.59 UserMenu.xaml

```
1 <!--
2  * Author    :    Ga l Mariot
3  * Project   :    Scrum'o'wall
4  * File      :    UserMenu.xaml
5  * Desc.     :    This file contains the basic template of the UserMenu view
6  -->
7 <Window x:Class="Scrum_o_wall.Views.UserMenu"
8       xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
9       xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
```

```

10    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
11    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
12    xmlns:local="clr-namespace:Scrum_o_wall.Views"
13    mc:Ignorable="d"
14    Title="Gestion des Utilisateurs" Height="450" Width="600" ←
15        Closing="UserMenu_Closing" WindowStartupLocation="CenterScreen" ←
16        FontSize="16" ResizeMode="NoResize">
17    <Canvas x:Name="cnvsStates" Margin="0">
18        <Grid x:Name="grdStates" Width="594" Height="421">
19            <Grid.ColumnDefinitions>
20                <ColumnDefinition Width="80*"/></ColumnDefinition>
21                <ColumnDefinition Width="20*"/></ColumnDefinition>
22                <ColumnDefinition Width="80*"/></ColumnDefinition>
23            </Grid.ColumnDefinitions>
24            <Grid.RowDefinitions>
25                <RowDefinition Height="7*"/></RowDefinition>
26                <RowDefinition Height="40*"/></RowDefinition>
27            </Grid.RowDefinitions>
28            <TextBlock Grid.ColumnSpan="3" FontSize="32" TextAlignment="Center" ←
29                VerticalAlignment="Center" >Gestion des Utilisateurs</TextBlock>
30        <GroupBox Margin="10" Header="Utilisateurs possibles" Grid.Row="1">
31            <Grid>
32                <Grid.RowDefinitions>
33                    <RowDefinition/></RowDefinition>
34                    <RowDefinition Height="70"/></RowDefinition>
35                </Grid.RowDefinitions>
36                <ListBox Margin="10" Name="lstPossibleUsers" ←
37                    MouseDoubleClick="Lst_MouseDoubleClick" ←
38                    Stylus.IsPressAndHoldEnabled="False">
39                </ListBox>
40                <Button TouchUp="BtnAddUser_Click" Click="BtnAddUser_Click" ←
41                    x:Name="btnAddUser" Margin="10" ←
42                    Grid.Row="1">Ajouter</Button>
43            </Grid>
44        </GroupBox>
45        <GroupBox Grid.Column="2" Margin="10" Header="Utilisateurs assignés" Grid.Row="1">
46            <ListBox Margin="10" Name="lstAssignedUsers" ←
47                MouseDoubleClick="Lst_MouseDoubleClick" ←
48                Stylus.IsPressAndHoldEnabled="False">
49            </ListBox>
50        </GroupBox>
51        <Grid Grid.Row="1" Grid.Column="1">
52            <Grid.RowDefinitions>
53                <RowDefinition/></RowDefinition>
54                <RowDefinition/></RowDefinition>
55                <RowDefinition/></RowDefinition>
56            </Grid.RowDefinitions>
57            <Button TouchUp="BtnGoLeft_Click" Click="BtnGoLeft_Click" ←
58                Margin="10" x:Name="btnGoLeft" Content="à " ></Button>
59            <Button TouchUp="BtnGoRight_Click" Click="BtnGoRight_Click" ←
60                Margin="10" x:Name="btnGoRight" Grid.Row="1" ←
61                Content="à " ></Button>
62            <Button TouchUp="BtnSave_Click" Click="BtnSave_Click" ←
63                Margin="10,10,10,28" x:Name="btnSave" Grid.Row="2" ←
64                Content=" " ></Button>
65        </Grid>
66    </Grid>
67 </Canvas>
68 </Window>

```

Listing 62 – ../../Scrum'o'wall/Scrum'o'wall/Views/UserMenu.xaml

2.60 UserMenu.xaml.cs

```

1  /*
2   * Author : Ga l Serge Mariot
3   * Project : Scrum'o'wall
4   * File : UserMenu.xaml.cs
5   * Desc. : This file contains the logic in the UserMenu view
6   */
7  using Scrum_o_wall.Classes;
8  using System;
9  using System.Collections.Generic;
10 using System.Windows;

```

```

11  using System.Windows.Controls;
12  using System.Windows.Input;
13
14 namespace Scrum_o_wall.Views
15 {
16     /// <summary>
17     /// Logique d'interaction pour UserMenu.xaml
18     /// </summary>
19     public partial class UserMenu : Window
20     {
21         private readonly Controller controller;
22         private readonly IUsersAssigned objectWithAssignedUsers;
23         private readonly List<User> possibleUsers;
24         public UserMenu(IUsersAssigned anObjectWithAssignedUsers, List<User> ↵
25             possibilities, Controller aController)
26         {
27             controller = aController;
28             objectWithAssignedUsers = anObjectWithAssignedUsers;
29             possibleUsers = possibilities;
30
31             InitializeComponent();
32
33             Refresh();
34         }
35
36         private void UserMenu_Closing(object sender, EventArgs e)
37         {
38             if (MessageBox.Show("Les changements non sauvegardés seront perdus.\nVoulez-vous sauvegarder les modifications?", "Attention", MessageBoxButton.YesNo, MessageBoxImage.Warning) == ↵
39             MessageBoxResult.Yes)
40             {
41                 Save();
42             }
43         }
44         private void Refresh()
45         {
46             lstAssignedUsers.Items.Clear();
47             lstPossibleUsers.Items.Clear();
48             foreach (User user in possibleUsers)
49             {
50                 if (objectWithAssignedUsers.GetUsers().Contains(user))
51                 {
52                     lstAssignedUsers.Items.Add(user);
53                 }
54                 else
55                 {
56                     lstPossibleUsers.Items.Add(user);
57                 }
58             }
59         }
60         private void BtnGoLeft_Click(object sender, EventArgs e)
61         {
62             if (lstAssignedUsers.SelectedItem is User state)
63             {
64                 lstAssignedUsers.Items.Remove(state);
65                 lstPossibleUsers.Items.Add(state);
66             }
67         }
68         private void BtnGoRight_Click(object sender, EventArgs e)
69         {
70             if (lstPossibleUsers.SelectedItem is User state)
71             {
72                 lstPossibleUsers.Items.Remove(state);
73                 lstAssignedUsers.Items.Add(state);
74             }
75         }
76         private void Save()
77         {
78             List<User> toRemove = new List<User>();
79             List<User> toAdd = new List<User>();
80             foreach (User user in controller.Users)
81             {
82                 if (lstAssignedUsers.Items.Contains(user) && ↵
83                     !objectWithAssignedUsers.GetUsers().Contains(user))
84                 {
85                     toAdd.Add(user);
86                 }
87             }
88         }
89     }
90 }

```

```

85             else if (lstPossibleUsers.Items.Contains(user) && ↵
86                 objectWithAssignedUsers.GetUsers().Contains(user))
87             {
88                 toRemove.Add(user);
89             }
90         foreach (User user in toAdd)
91         {
92             controller.AddUserToIUsersAssigned(user, objectWithAssignedUsers);
93         }
94         foreach (User user in toRemove)
95         {
96             controller.RemoveUserFromIUsersAssigned(user, ↵
97                 objectWithAssignedUsers);
98         }
99     }
100    private void BtnSave_Click(object sender, EventArgs e)
101    {
102        Save();
103        Close();
104    }
105    private void BtnAddUser_Click(object sender, EventArgs e)
106    {
107        UserCreate userCreate = new UserCreate();
108        if (userCreate.ShowDialog() == true)
109        {
110            controller.CreateUser(userCreate.tbxUserName.Text, ↵
111                objectWithAssignedUsers);
112            Refresh();
113        }
114    }
115    private void Lst_MouseDoubleClick(object sender, MouseButtonEventArgs e)
116    {
117        User user = (sender as ListBox).SelectedItem as User;
118        UserEdit userEdit = new UserEdit(user);
119        if (userEdit.ShowDialog() == true)
120        {
121            if (userEdit.Deleted)
122            {
123                controller.Delete(user);
124            }
125            else
126            {
127                string name = userEdit.tbxUserName.Text.Trim();
128                controller.UpdateUser(name, user);
129            }
130            Refresh();
131        }
132    }
133}

```

Listing 63 – ../../Scrum'o'wall/Scrum'o'wall/Views/UserMenu.xaml.cs

2.61 UserStoryCreate.xaml

```

1<!--
2 * Author      :   Gaël Mariot
3 * Project     :   Scrum'o'wall
4 * File        :   UserStoryCreate.xaml
5 * Desc.       :   This file contains the basic template of the UserStoryCreate view
6 -->
7<Window x:Class="Scrum_o_wall.Views.UserStoryCreate"
8     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
9     xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
10    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
11    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
12    xmlns:local="clr-namespace:Scrum_o_wall.Views"
13    mc:Ignorable="d"
14    Title="Création d'une User Story" Height="491" Width="508.713" ↵
15        WindowStartupLocation="CenterScreen" FontSize="16" >
16    <Grid>
17        <Grid.ColumnDefinitions>
18            <ColumnDefinition Width="163*"/>
19            <ColumnDefinition Width="226*"/>
20        </Grid.ColumnDefinitions>

```

```

20      <Grid.RowDefinitions>
21          <RowDefinition Height="56*"/>
22          <RowDefinition Height="148*"/>
23          <RowDefinition Height="44*"/>
24          <RowDefinition Height="42*"/>
25          <RowDefinition Height="42*"/>
26          <RowDefinition Height="42*"/>
27          <RowDefinition Height="45*"/>
28      </Grid.RowDefinitions>
29      <Label Grid.ColumnSpan="2" HorizontalAlignment="Center" ←
30          VerticalAlignment="Center" FontSize="36" Height="58" ←
31          Margin="44,-2,44,0" Width="336" >Cr er une User Story</Label>
32
33      <Label Grid.Row="1" HorizontalAlignment="Right" VerticalAlignment="Top" ←
34          >Description de la User Story:</Label>
35      <TextBox Grid.Row="1" TextWrapping="Wrap" MaxLength="65535" ←
36          Name="txbDesc" Grid.Column="1" Margin="10" ←
37          Stylus.IsPressAndHoldEnabled="False"/>
38
39      <Label Grid.Row="2" HorizontalAlignment="Right" ←
40          VerticalAlignment="Center" >Date limite:</Label>
41      <DatePicker x:Name="dtpckrDateLimit" Grid.Column="1" Margin="10" ←
42          Grid.Row="2" Stylus.IsPressAndHoldEnabled="False"/>
43
44      <Label Grid.Row="3" HorizontalAlignment="Right" ←
45          VerticalAlignment="Center" >Estimation de complexit :</Label>
46      <TextBox Grid.Row="3" Name="txbComplexity" MaxLength="5" ←
47          Grid.Column="1" KeyDown="TbxComplexity_KeyDown" Margin="10" ←
48          Stylus.IsPressAndHoldEnabled="False"/>
49
50  </Grid>
51 </Window>

```

Listing 64 – ../../Scrum'o'wall/Scrum'o'wall/Views/UserStoryCreate.xaml

2.62 UserStoryCreate.xaml.cs

```

1  /*
2   * Author    :  Ga l Serge Mariot
3   * Project   :  Scrum'o'wall
4   * File      :  UserStoryCreate.xaml.cs
5   * Desc.     :  This file contains the logic in the UserStoryCreate view
6   */
7  using Scrum_o_wall.Classes;
8  using System;
9  using System.Windows;
10 using System.Windows.Input;
11
12 namespace Scrum_o_wall.Views
13 {
14     /// <summary>
15     /// Logique d'interaction pour UserStoryCreate.xaml
16     /// </summary>
17     public partial class UserStoryCreate : Window
18     {
19         public UserStoryCreate(Controller aController)
20         {
21             InitializeComponent();
22
23             foreach (Priority priority in aController.Priorities)
24             {
25                 cbxPriority.Items.Add(priority);
26             }
27         }
28
29         private void btnConfirm_Click(object sender, RoutedEventArgs e)
30         {
31             // Logic to handle confirmation click
32         }
33
34         private void btnCancel_Click(object sender, RoutedEventArgs e)
35         {
36             // Logic to handle cancel click
37         }
38     }
39 }

```

```

26     }
27     foreach (Classes.Type type in aController.Types)
28     {
29         cbxType.Items.Add(type);
30     }
31 }
32
33     private void TbxComplexity_KeyDown(object sender, KeyEventArgs e)
34     {
35         bool isNumber = (e.Key >= Key.D0 && e.Key <= Key.D9);
36         if (!isNumber)
37         {
38             e.Handled = true;
39         }
40     }
41     private void BtnCancel_Click(object sender, EventArgs e)
42     {
43         Close();
44     }
45     private void BtnConfirm_Click(object sender, EventArgs e)
46     {
47         int descLength = tbxDesc.Text.Trim().Length;
48         int complexityLength = tbxComplexity.Text.Trim().Length;
49         if (descLength > 0 && complexityLength > 0 && ←
50             cbxPriority.SelectedIndex >= 0 && cbxType.SelectedIndex >= 0)
51         {
52             DialogResult = true;
53             Close();
54         }
55         else
56         {
57             MessageBox.Show("Un ou plusieurs champ(s) n'est pas rempli!", ←
58                             "Erreur", MessageBoxButtons.OK, MessageBoxIcon.Error);
59         }
60     }
61 }

```

Listing 65 – ../../Scrum'o'wall/Scrum'o'wall/Views/UserStoryCreate.xaml.cs

2.63 UserStoryEdit.xaml

```

1 <!--
2 * Author : Ga l Mariot
3 * Project : Scrum'o'wall
4 * File : UserStoryEdit.xaml
5 * Desc. : This file contains the basic template of the UserStoryEdit view
6 -->
7 <Window x:Class="Scrum_o_wall.Views.UserStoryEdit"
8     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
9     xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
10    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
11    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
12    xmlns:local="clr-namespace:Scrum_o_wall.Views"
13    mc:Ignorable="d"
14    Title="Modification d'une User Story" Height="722.912" Width="643.787" ←
15        WindowStartupLocation="CenterScreen" FontSize="16">
16     <Grid>
17         <Grid.ColumnDefinitions>
18             <ColumnDefinition Width="163*"/>
19             <ColumnDefinition Width="226*"/>
20         </Grid.ColumnDefinitions>
21         <Grid.RowDefinitions>
22             <RowDefinition Height="55*"/>
23             <RowDefinition Height="146*"/>
24             <RowDefinition Height="44*"/>
25             <RowDefinition Height="41*"/>
26             <RowDefinition Height="41*"/>
27             <RowDefinition Height="41*"/>
28             <RowDefinition Height="41*"/>
29             <RowDefinition Height="41*"/>
30             <RowDefinition Height="41*"/>
31             <RowDefinition Height="44*"/>
32         </Grid.RowDefinitions>

```

```

33     <Label Grid.ColumnSpan="2" HorizontalAlignment="Center" ←
34         VerticalAlignment="Center" FontSize="36" >Modification d'une User ←
35             Story</Label>
36
37     <Label Grid.Row="1" HorizontalAlignment="Right" ←
38         VerticalAlignment="Top">Description de la User Story :</Label>
39     <TextBox Grid.Row="1" TextWrapping="Wrap" MaxLength="65535" ←
40         Name="txbDesc" Grid.Column="1" Margin="10" ←
41             Stylus.IsPressAndHoldEnabled="False"/>
42
43     <Label Grid.Row="2" HorizontalAlignment="Right" ←
44         VerticalAlignment="Center" >Date limite :</Label>
45     <DatePicker x:Name="dtpckrDateLimit" Grid.Column="1" Margin="10" ←
46         Grid.Row="2" Stylus.IsPressAndHoldEnabled="False"/>
47
48     <Label Grid.Row="3" HorizontalAlignment="Right" ←
49         VerticalAlignment="Center" >Estimation de complexité :</Label>
50     <TextBox Grid.Row="3" Name="txbComplexity" MaxLength="5" ←
51         Grid.Column="1" KeyDown="TbxComplexity_KeyDown" Margin="10" ←
52             Stylus.IsPressAndHoldEnabled="False"/>
53
54     <Label Grid.Row="4" HorizontalAlignment="Right" ←
55         VerticalAlignment="Center" >Complexité accomplie:</Label>
56     <TextBox Grid.Row="4" Name="txbCompletedComplexity" MaxLength="5" ←
57         Grid.Column="1" KeyDown="TbxCompletedComplexity_KeyDown" Margin="10" ←
58             Stylus.IsPressAndHoldEnabled="False"/>
59
60     <Label Grid.Row="5" HorizontalAlignment="Right" ←
61         VerticalAlignment="Center" >Priorité :</Label>
62     <ComboBox Name="cbxPriority" Grid.Row="5" Margin="10" Grid.Column="1" ←
63         Stylus.IsPressAndHoldEnabled="False"/></ComboBox>
64
65     <Label Grid.Row="6" HorizontalAlignment="Right" ←
66         VerticalAlignment="Center" >Type :</Label>
67     <ComboBox Name="cbxType" Grid.Row="6" Margin="10" Grid.Column="1" ←
68         Stylus.IsPressAndHoldEnabled="False"/></ComboBox>
69
70     <CheckBox Grid.Row="7" x:Name="chckBxBlocked" ←
71         HorizontalAlignment="Center" VerticalAlignment="Center" ←
72         Grid.ColumnSpan="2" ←
73             Stylus.IsPressAndHoldEnabled="False">Bloqué </CheckBox>
74
75     <Grid Grid.Row="8" Grid.ColumnSpan="2">
76         <Grid.ColumnDefinitions>
77             <ColumnDefinition></ColumnDefinition>
78             <ColumnDefinition></ColumnDefinition>
79             <ColumnDefinition></ColumnDefinition>
80         </Grid.ColumnDefinitions>
81         <Button TouchUp="BtnFiles_Click" Grid.Column="0" x:Name="btnFiles" ←
82             Click="BtnFiles_Click" Margin="10">Fichiers</Button>
83         <Button TouchUp="BtnComments_Click" Grid.Column="1" ←
84             x:Name="btnComments" Click="BtnComments_Click" ←
85             Margin="10">Commentaires</Button>
86         <Button TouchUp="BtnChecklists_Click" Grid.Column="2" ←
87             x:Name="btnChecklists" Click="BtnChecklists_Click" ←
88             Margin="10">Checklists</Button>
89     </Grid>
90
91     <Grid Grid.Row="9" Grid.ColumnSpan="2">
92         <Grid.ColumnDefinitions>
93             <ColumnDefinition></ColumnDefinition>
94             <ColumnDefinition></ColumnDefinition>
95         </Grid.ColumnDefinitions>
96         <Button TouchUp="BtnActivities_Click" Grid.Column="0" ←
97             x:Name="btnActivities" Click="BtnActivities_Click" ←
98             Margin="10">Activités</Button>
99         <Button TouchUp="BtnUserAssigned_Click" Grid.Column="1" ←
100            x:Name="btnUserAssigned" Click="BtnUserAssigned_Click" ←
101            Margin="10">Utilisateurs assignés</Button>
102     </Grid>
103
104     <Grid Grid.Row="10" Grid.ColumnSpan="2">
105         <Grid.ColumnDefinitions>
106             <ColumnDefinition></ColumnDefinition>
107             <ColumnDefinition></ColumnDefinition>
108             <ColumnDefinition Width="2*"></ColumnDefinition>
109         </Grid.ColumnDefinitions>
110         <Button x:Name="btnCancel" Margin="10" Click="BtnCancel_Click" ←
111             TouchUp="BtnCancel_Click">Annuler</Button>

```

```

81      <Button x:Name="btnDelete" Grid.Column="1" Margin="10" ↵
82          Click="BtnDelete_Click" ↵
83          TouchUp="BtnDelete_Click">Supprimer</Button>
84      <Button x:Name="btnConfirm" Grid.Column="2" Margin="10" ↵
85          Click="BtnConfirm_Click" ↵
86          TouchUp="BtnConfirm_Click">Confirmer</Button>
87  
```

Listing 66 – ../../Scrum'o'wall/Scrum'o'wall/Views/UserStoryEdit.xaml

2.64 UserStoryEdit.xaml.cs

```

1  /*
2   * Author    : Gaël Serge Mariot
3   * Project   : Scrum'o'wall
4   * File      : UserStoryEdit.xaml.cs
5   * Desc.     : This file contains the logic in the UserStoryEdit view
6   */
7  using Scrum_o_wall.Classes;
8  using System;
9  using System.Windows;
10 using System.Windows.Input;
11
12 namespace Scrum_o_wall.Views
13 {
14     ///<summary>
15     /// Logique d'interaction pour UserStoryEdit.xaml
16     ///</summary>
17     public partial class UserStoryEdit : Window
18     {
19         private readonly UserStory userStory;
20         private readonly Controller controller;
21         private readonly Project project;
22         public bool Deleted = false;
23
24         public UserStoryEdit(UserStory aUserStory, Project aProject, Controller ←
25                           aController)
26         {
27             InitializeComponent();
28             userStory = aUserStory;
29             controller = aController;
30             project = aProject;
31
32             tbxDesc.Text = userStory.Description;
33             dtprkrDateLimit.SelectedDate = userStory.DateLimit;
34             tbxComplexity.Text = userStory.ComplexityEstimation.ToString();
35             tbxCompletedComplexity.Text = ←
36                 userStory.CompletedComplexity.ToString();
37
38             foreach (Priority p in controller.Priorities)
39             {
40                 cbxPriority.Items.Add(p);
41             }
42             foreach (Classes.Type t in controller.Types)
43             {
44                 cbxType.Items.Add(t);
45             }
46
47             cbxPriority.SelectedItem = userStory.Priority;
48             cbxType.SelectedItem = userStory.Type;
49
50             chckBxBlocked.IsChecked = userStory.Blocked;
51         }
52
53         private void TbxComplexity_KeyDown(object sender, KeyEventArgs e)
54         {
55             bool isNumber = (e.Key >= Key.D0 && e.Key <= Key.D9);
56             if (!isNumber)
57             {
58                 e.Handled = true;
59             }
60         }
61         private void BtnActivities_Click(object sender, EventArgs e)
62         {
63             ...
64         }
65     }
66 }
```

```

62         ActivitiesMenu activitiesMenu = new ↪
63             ActivitiesMenu(userStory.Activities);
64         activitiesMenu.ShowDialog();
65     }
66     private void BtnUserAssigned_Click(object sender, EventArgs e)
67     {
68         UserMenu userMenu = new UserMenu(userStory, project.GetUsers(), ↪
69             controller);
70         userMenu.ShowDialog();
71     }
72     private void BtnChecklists_Click(object sender, EventArgs e)
73     {
74         ChecklistMenu checklistMenu = new ChecklistMenu(userStory, ↪
75             controller);
76         checklistMenu.ShowDialog();
77     }
78     private void BtnComments_Click(object sender, EventArgs e)
79     {
80         CommentMenu commentMenu = new CommentMenu(userStory, controller);
81         commentMenu.ShowDialog();
82     }
83     private void BtnFiles_Click(object sender, EventArgs e)
84     {
85         FileMenu fileMenu = new FileMenu(userStory, controller);
86         fileMenu.ShowDialog();
87     }
88     private void BtnConfirm_Click(object sender, EventArgs e)
89     {
90         int descLength = tbxDesc.Text.Trim().Length;
91         int complexityLength = tbxComplexity.Text.Trim().Length;
92         int completedLength = tbxCompletedComplexity.Text.Trim().Length;
93         if (completedLength > 0 && complexityLength > 0 && descLength > 0)
94         {
95             DialogResult = true;
96             Close();
97         }
98         else
99         {
100             MessageBox.Show("Un ou plusieurs champ(s) n'est pas rempli!", ↪
101                 "Erreur", MessageBoxButtons.OK, MessageBoxIcon.Error);
102         }
103     }
104     private void btnCancel_Click(object sender, EventArgs e)
105     {
106         Close();
107     }
108     private void BtnDelete_Click(object sender, EventArgs e)
109     {
110         if (MessageBox.Show("La user story sera supprim e.\n\t- vous\t<→
111             s r(e)?", "Attention", MessageBoxButtons.YesNo, ↪
112             MessageBoxIcon.Warning) == DialogResult.Yes)
113         {
114             DialogResult = true;
115             Deleted = true;
116             Close();
117         }
118     }
119 }

```

Listing 67 – ../../Scrum'o'wall/Scrum'o'wall/Views/UserStoryEdit.xaml.cs

3 Modèles

3.1 Activity.cs

```
1  /*
2   * Author    : Gaël Serge Mariot
3   * Project   : Scrum'o'wall
4   * File      : Activity.cs
5   * Desc.     : This file contains the structure of the Activity class
6   */
7  using System;
8
9 namespace Scrum_o_wall.Classes
10 {
11     public class Activity
12     {
13         private readonly int id;
14         private int userStoryId;
15         private UserStory userStory;
16
17         public Activity(int id, string description, DateTime dateTime, int ↪
18                         userStoryId)
19         {
20             this.id = id;
21             Description = description;
22             DateTime = dateTime;
23             this.userStoryId = userStoryId;
24         }
25
26         public int Id => id;
27         public string Description { get; set; }
28         public DateTime DateTime { get; set; }
29         public int UserStoryId => userStoryId;
30         public UserStory UserStory
31         {
32             get => userStory; set
33             {
34                 userStory = value;
35                 userStoryId = value.Id;
36             }
37         }
38
39         public override string ToString()
40         {
41             return DateTime.ToString("dd.MM.yyyy - HH:mm:ss") + " - " + ↪
42             Description;
43         }
44     }
45 }
```

Listing 68 – ../../Scrum'o'wall/Scrum'o'wall/Classes/Activity.cs

3.2 Checklist.cs

```
1  /*
2   * Author    : Gaël Serge Mariot
3   * Project   : Scrum'o'wall
4   * File      : Checklist.cs
5   * Desc.     : This file contains the structure of the Checklist class
6   */
7  using System.Collections.Generic;
8
9 namespace Scrum_o_wall.Classes
10 {
11     public class Checklist
12     {
13         private readonly int id;
14         private int userStoryId;
15         private UserStory userStory;
16
17         public Checklist(int id, string name, int userStoryId)
18         {
19             this.id = id;
20             Name = name;
21             this.userStoryId = userStoryId;
22         }
23     }
24 }
```

```

23
24     public int Id => id;
25     public string Name { get; set; }
26     public int UserStoryId => userStoryId;
27     public UserStory UserStory
28     {
29         get => userStory; set
30         {
31             userStory = value;
32             userStoryId = value.Id;
33         }
34     }
35     public List<ChecklistItem> ChecklistItems { get; set; } = new ←
36         List<ChecklistItem>();
37     public override string ToString()
38     {
39         return Name;
40     }
41 }
```

Listing 69 – ../../Scrum'o'wall/Scrum'o'wall/Classes/Checklist.cs

3.3 ChecklistItem.cs

```

1  /*
2  * Author    :   Gael Serge Mariot
3  * Project   :   Scrum'o'wall
4  * File      :   ChecklistItem.cs
5  * Desc.     :   This file contains the structure of the ChecklistItem class
6  */
7  using System.Collections.Generic;
8
9 namespace Scrum_o_wall.Classes
10{
11    public class ChecklistItem : IUsersAssigned
12    {
13        private readonly int id;
14        private int checklistId;
15        private Checklist checklist;
16        private readonly List<User> assignedUsers = new List<User>();
17
18        public ChecklistItem(int id, string nameItem, bool done, int checklistId)
19        {
20            this.id = id;
21            NameItem = nameItem;
22            Done = done;
23            this.checklistId = checklistId;
24        }
25
26        public int Id => id;
27        public string NameItem { get; set; }
28        public bool Done { get; set; }
29        public int ChecklistId => checklistId;
30        public Checklist Checklist
31        {
32            get => checklist; set
33            {
34                checklist = value;
35                checklistId = value.Id;
36            }
37        }
38
39        public void AddUser(User user)
40        {
41            assignedUsers.Add(user);
42        }
43
44        public List<User> GetUsers()
45        {
46            return assignedUsers;
47        }
48
49        public void RemoveUser(User user)
50        {
51            assignedUsers.Remove(user);
52        }
53 }
```

```

53     public override string ToString()
54     {
55         return NameItem;
56     }
57 }
58

```

Listing 70 – ../../Scrum'o'wall/Scrum'o'wall/Classes/ChecklistItem.cs

3.4 Comment.cs

```

1  /*
2  * Author    :   Ga l Serge Mariot
3  * Project   :   Scrum'o'wall
4  * File      :   Comment.cs
5  * Desc.     :   This file contains the structure of the Comment class
6  */
7  using System;
8
9 namespace Scrum_o_wall.Classes
10{
11    public class Comment
12    {
13        private readonly int id;
14        private int userStoryId;
15        private int userId;
16        private UserStory userStory;
17        private User user;
18
19        public Comment(int id, string description, DateTime dateTIme, int ↵
20                      userStoryId, int userId)
21        {
22            this.id = id;
23            Description = description;
24            DateTIme = dateTIme;
25            this.userStoryId = userStoryId;
26            this.userId = userId;
27        }
28
29        public int Id => id;
30        public string Description { get; set; }
31        public DateTime DateTIme { get; set; }
32        public int UserStoryId => userStoryId;
33        public UserStory UserStory
34        {
35            get => userStory; set
36            {
37                userStory = value;
38                userStoryId = value.Id;
39            }
40        }
41        public int UserId => userId;
42        public User User
43        {
44            get => user;
45            set
46            {
47                user = value;
48                userId = value.Id;
49            }
50        }
51        public override string ToString()
52        {
53            return User.Name + " - " + Description;
54        }
55    }
56}
57

```

Listing 71 – ../../Scrum'o'wall/Scrum'o'wall/Classes/Comment.cs

3.5 File.cs

```

1  /*
2  * Author    :   Ga l Serge Mariot

```

```

3 * Project    : Scrum'o'wall
4 * File       : File.cs
5 * Desc.      : This file contains the structure of the TaskFile class
6 */
7
8 namespace Scrum_o_wall.Classes
9 {
10    public class File
11    {
12        private readonly int id;
13        private int userStoryId;
14        private UserStory userStory;
15
16
17        public File(int id, string name, string description, int userStoryId)
18        {
19            this.id = id;
20            Name = name;
21            Description = description;
22            this.userStoryId = userStoryId;
23        }
24
25        public int Id => id;
26        public string Name { get; set; }
27        public string Description { get; set; }
28        public UserStory UserStory
29        {
30            get => userStory; set
31            {
32                userStory = value;
33                userStoryId = value.Id;
34            }
35        }
36        public int UserStoryId => userStoryId;
37
38        public override string ToString()
39        {
40            return Description;
41        }
42    }
43}

```

Listing 72 – ../../Scrum'o'wall/Scrum'o'wall/Classes/File.cs

3.6 IUsersAssigned.cs

```

1 /*
2 * Author     : Ga l Serge Mariot
3 * Project   : Scrum'o'wall
4 * File      : IUsersAssigned.cs
5 * Desc.     : This file is the interface to some classes with user assigned
6 */
7 using System.Collections.Generic;
8
9 namespace Scrum_o_wall.Classes
10 {
11    public interface IUsersAssigned
12    {
13        List<User> GetUsers();
14        void AddUser(User user);
15        void RemoveUser(User user);
16    }
17 }
18

```

Listing 73 – ../../Scrum'o'wall/Scrum'o'wall/Classes/IUsersAssigned.cs

3.7 MindMap.cs

```

1 /*
2 * Author     : Ga l Serge Mariot
3 * Project   : Scrum'o'wall
4 * File      : MindMap.cs
5 * Desc.     : This file contains the structure of the MindMap class

```

```

6  */
7  using System.Collections.Generic;
8
9 namespace Scrum_o_wall.Classes
10 {
11     public class MindMap
12     {
13         private readonly int id;
14         private int projectId;
15         private Project project;
16         public MindMap(int anId, string aName, int aProjectId)
17         {
18             id = anId;
19             Name = aName;
20             projectId = aProjectId;
21         }
22
23         public Node Root { get; set; }
24         public string Name { get; set; }
25         public int Id => id;
26         public int ProjectId => projectId;
27         public Project Project
28         {
29             get => project;
30             set
31             {
32                 project = value;
33                 projectId = value.Id;
34             }
35         }
36         public List<Node> GetAllNodes()
37         {
38             return Root.AllChildrens();
39         }
40         public override string ToString()
41         {
42             return Name;
43         }
44     }
45 }

```

Listing 74 – ../../Scrum'o'wall/Scrum'o'wall/Classes/MindMap.cs

3.8 Node.cs

```

1 /*
2 * Author : Gaël Mariot
3 * Project : Scrum'o'wall
4 * File : Node.cs
5 * Desc. : This file contains the structure of the Node class
6 */
7 using System.Collections.Generic;
8
9 namespace Scrum_o_wall.Classes
10 {
11     public class Node
12     {
13
14         private readonly int id;
15         private Node previous;
16         private int? previousId;
17         private int mindmapId;
18         private MindMap mindMap;
19
20         public Node(int anId, string aName, int? aPreviousId, int aMindmapId)
21         {
22             id = anId;
23             Name = aName;
24             previousId = aPreviousId;
25             mindmapId = aMindmapId;
26         }
27
28         public int Id => id;
29         public string Name { get; set; }
30         public int? PreviousId => previousId;
31         public int Level
32         {

```

```

33     get
34     {
35         int lvl = 0;
36         Node n = this;
37         while (n.Previous != null)
38         {
39             n = n.Previous;
40             lvl++;
41         }
42         return lvl;
43     }
44 }
45 public Node Previous
46 {
47     get => previous;
48     set
49     {
50         previous = value;
51         if (value == null)
52         {
53             previousId = null;
54         }
55         else
56         {
57             previousId = value.Id;
58         }
59     }
60 }
61 public int MindmapId => mindmapId;
62 public MindMap MindMap
63 {
64     get => mindMap;
65     set
66     {
67         mindMap = value;
68         mindmapId = value.Id;
69     }
70 }
71 public List<Node> Childrens { get; set; } = new List<Node>();
72 public List<Node> AllChildrens(List<Node> aList = null)
73 {
74     if (aList == null)
75     {
76         aList = new List<Node>
77         {
78             this
79         };
80     }
81     aList.AddRange(Childrens);
82     foreach (Node n in Childrens)
83     {
84         n.AllChildrens(aList);
85     }
86     return aList;
87 }
88 public override string ToString()
89 {
90     return Name;
91 }
92 }
93 }
94 }
```

Listing 75 – ../../Scrum'o'wall/Scrum'o'wall/Classes/Node.cs

3.9 Priority.cs

```

1  /*
2  * Author    :   Ga l Serge Mariot
3  * Project   :   Scrum'o'wall
4  * File      :   Priority.cs
5  * Desc.     :   This file contains the structure of the Priority class
6  */
7
8 namespace Scrum_o_wall.Classes
9 {
10    public class Priority
```

```

11     {
12         private readonly int id;
13
14         public Priority(int id, string name)
15         {
16             this.id = id;
17             Name = name;
18         }
19
20         public int Id => id;
21         public string Name { get; set; }
22         public override string ToString()
23         {
24             return Name;
25         }
26     }
27 }
```

Listing 76 – ../../Scrum'o'wall/Scrum'o'wall/Classes/Priority.cs

3.10 Project.cs

```

1  /*
2   * Author    :    Gael Serge Mariot
3   * Project   :    Scrum'o'wall
4   * File      :    Project.cs
5   * Desc.     :    This file contains the structure of the Project class
6   */
7  using System;
8  using System.Collections.Generic;
9
10 namespace Scrum_o_wall.Classes
11 {
12     public class Project : IUsersAssigned
13     {
14         private readonly int id;
15         private readonly List<User> assignedUsers = new List<User>();
16
17         /// <summary>
18         /// Create a project with name,description and date
19         /// </summary>
20         /// <param name="aName">the name of the project</param>
21         /// <param name="aDesc">the description of the project</param>
22         /// <param name="aBegin">the date of beginning of project</param>
23         public Project(int anId, string aName, string aDesc, DateTime aBegin)
24         {
25             id = anId;
26             Name = aName;
27             Description = aDesc;
28             Begin = aBegin;
29         }
30
31         //properties declaration
32         public int Id => id;
33         public DateTime Begin { get; set; }
34         public string Name { get; set; }
35         public string Description { get; set; }
36         public List<UserStory> AllUserStories { get; set; } = new ←
37             List<UserStory>();
38         public List<Sprint> Sprints { get; set; } = new List<Sprint>();
39         public List<MindMap> MindMaps { get; set; } = new List<MindMap>();
40         public Dictionary<int, State> States { get; set; } = new ←
41             Dictionary<int, State>();
42
43         public void AddUser(User user)
44         {
45             assignedUsers.Add(user);
46         }
47
48         public List<User> GetUsers()
49         {
50             return assignedUsers;
51         }
52
53         public void RemoveUser(User user)
54         {
```

```

54         assignedUsers.Remove(user);
55     }
56
57     public override string ToString()
58     {
59         return string.Format("Project {0}: {1} - {2} - {3}", Id, ←
60             Name, Begin, Description);
61     }
62 }
63

```

Listing 77 – ../../Scrum'o'wall/Scrum'o'wall/Classes/Project.cs

3.11 Sprint.cs

```

1  /*
2  * Author    :   Gaël Serge Mariot
3  * Project   :   Scrum'o'wall
4  * File      :   Sprint.cs
5  * Desc.     :   This file contains the structure of the Sprint class
6  */
7 using System;
8 using System.Collections.Generic;
9
10 namespace Scrum_o_wall.Classes
11 {
12     public class Sprint
13     {
14         private readonly int id;
15         private int projectId;
16         private Project project;
17
18         public Sprint(int anId, DateTime aBegin, DateTime anEnd, int aprojectId)
19         {
20             id = anId;
21             Begin = aBegin;
22             End = anEnd;
23             projectId = aprojectId;
24         }
25         public int Id => id;
26         public DateTime Begin { get; set; }
27         public DateTime End { get; set; }
28         public int ProjectId => projectId;
29         public Project Project
30         {
31             get => project;
32             set
33             {
34                 project = value;
35                 projectId = value.Id;
36             }
37         }
38         public Dictionary<int, UserStory> OrderedUserStories { get; } = new ←
39             Dictionary<int, UserStory>();
40
41         #region Add/Remove userStories
42         public void AddUserStory(int order, UserStory toAdd)
43         {
44             OrderedUserStories.Add(order, toAdd);
45         }
46         public void AddListUserStories(Dictionary<int, UserStory> ListToAdd)
47         {
48             foreach (KeyValuePair<int, UserStory> item in ListToAdd)
49             {
50                 OrderedUserStories.Add(item.Key, item.Value);
51             }
52         }
53         public void RemoveUserStoryByOrder(int order)
54         {
55             OrderedUserStories.Remove(order);
56         }
57         #endregion
58
59         public override string ToString()
60         {

```

```

60             return string.Format("Sprint du {0} au {1}", ↵
61                 Begin.ToShortDateString(), End.ToShortDateString());
62         }
63     }

```

Listing 78 – ../../Scrum'o'wall/Scrum'o'wall/Classes/Sprint.cs

3.12 State.cs

```

1 /*
2  * Author    :   Ga l Serge Mariot
3  * Project   :   Scrum'o'wall
4  * File      :   State.cs
5  * Desc.     :   This file contains the structure of the State class
6 */
7
8 namespace Scrum_o_wall.Classes
9 {
10    public class State
11    {
12        private readonly int id;
13
14        public State(int id, string name)
15        {
16            this.id = id;
17            Name = name;
18        }
19
20        public int Id => id;
21        public string Name { get; set; }
22        public override string ToString()
23        {
24            return Name;
25        }
26    }
27 }

```

Listing 79 – ../../Scrum'o'wall/Scrum'o'wall/Classes/State.cs

3.13 Type.cs

```

1 /*
2  * Author    :   Ga l Serge Mariot
3  * Project   :   Scrum'o'wall
4  * File      :   Type.cs
5  * Desc.     :   This file contains the structure of the Type class
6 */
7
8 namespace Scrum_o_wall.Classes
9 {
10    public class Type
11    {
12        private readonly int id;
13
14        public Type(int id, string name)
15        {
16            this.id = id;
17            Name = name;
18        }
19
20        public int Id => id;
21        public string Name { get; set; }
22        public override string ToString()
23        {
24            return Name;
25        }
26    }
27 }

```

Listing 80 – ../../Scrum'o'wall/Scrum'o'wall/Classes/Type.cs

3.14 User.cs

```
1  /*
2   * Author    : Ga l Serge Mariot
3   * Project   : Scrum'o'wall
4   * File      : User.cs
5   * Desc.     : This file contains the structure of the User class
6   */
7
8  namespace Scrum_o_wall.Classes
9 {
10    public class User
11    {
12      private readonly int id;
13
14      public User(int id, string name)
15      {
16        this.id = id;
17        Name = name;
18      }
19
20      public int Id => id;
21      public string Name { get; set; }
22      public override string ToString()
23      {
24        return Name;
25      }
26    }
27 }
```

Listing 81 – ../../Scrum'o'wall/Scrum'o'wall/Classes/User.cs

3.15 UserStory.cs

```
1  /*
2   * Author    : Ga l Serge Mariot
3   * Project   : Scrum'o'wall
4   * File      : UserStory.cs
5   * Desc.     : This file contains the structure of the UserStory class
6   */
7  using System;
8  using System.Collections.Generic;
9
10 namespace Scrum_o_wall.Classes
11 {
12   public class UserStory : IUsersAssigned
13   {
14     private readonly int id;
15     private int stateId;
16     private int projectId;
17     private int typeId;
18     private int priorityId;
19     private readonly List<User> assignedUsers = new List<User>();
20     private Type type;
21     private Priority priority;
22     private State state;
23     private Project project;
24
25     public UserStory(int anId, string aDesc, DateTime? aDateLimit, int ↵
26                      aComplexity, int aCompletedComplexity, bool isBlocked, int ↵
27                      aProjectId, int aStateId, int aTypeId, int aPriorityId)
28     {
29       id = anId;
30       Description = aDesc;
31       DateLimit = aDateLimit;
32       ComplexityEstimation = aComplexity;
33       CompletedComplexity = aCompletedComplexity;
34       Blocked = isBlocked;
35       projectId = aProjectId;
36       stateId = aStateId;
37       typeId = aTypeId;
38       priorityId = aPriorityId;
39     }
40
41     public int Id => id;
42     public int StateId => stateId;
```

```

41     public int ProjectId => projectId;
42     public int TypeId => typeId;
43     public int PriorityId => priorityId;
44     public string Description { get; set; }
45     public State State
46     {
47         get => state;
48         set
49         {
50             state = value;
51             stateId = value.Id;
52         }
53     }
54     public DateTime? DateLimit { get; set; }
55     public int CompletedComplexity { get; set; }
56     public int ComplexityEstimation { get; set; }
57     public bool Blocked { get; set; }
58     public Type Type
59     {
60         get => type;
61         set
62         {
63             type = value;
64             typeId = value.Id;
65         }
66     }
67     public Project Project
68     {
69         get => project;
70         set
71         {
72             project = value;
73             projectId = value.Id;
74         }
75     }
76     public List<File> Files { get; set; } = new List<File>();
77     public List<Comment> Comments { get; set; } = new List<Comment>();
78     public List<Activity> Activities { get; set; } = new List<Activity>();
79     public List<Checklist> Checklists { get; set; } = new List<Checklist>();
80     public Priority Priority
81     {
82         get => priority;
83         set
84         {
85             priority = value;
86             priorityId = value.Id;
87         }
88     }
89
90
91     public void AddUser(User user)
92     {
93         assignedUsers.Add(user);
94     }
95
96     public List<User> GetUsers()
97     {
98         return assignedUsers;
99     }
100
101    public void RemoveUser(User user)
102    {
103        assignedUsers.Remove(user);
104    }
105    public override string ToString()
106    {
107        return Description;
108    }
109}
110}

```

Listing 82 – ../../Scrum'o'wall/Scrum'o'wall/Classes/UserStory.cs

4 Contrôleurs

4.1 Controller.cs

```
1  /*
2   * Author    :   Ga l Serge Mariot
3   * Project   :   Scrum'o'wall
4   * File      :   Controller.cs
5   * Desc.     :   This file is the control class. It is used by the view to ←
6   *                 communicate with the database.
7 */
8 using Scrum_o_wall.Classes;
9 using System;
10 using System.Collections.Generic;
11 using System.Linq;
12
13 namespace Scrum_o_wall
14 {
15     public class Controller
16     {
17         private List<Project> projects;
18         private List<User> users;
19
20         private List<Classes.Type> types;
21         private List<Priority> priorities;
22         private List<State> states;
23
24         public List<Project> Projects => projects;
25         public List<User> Users => users;
26         public List<Classes.Type> Types => types;
27         public List<Priority> Priorities => priorities;
28         public List<State> States => states;
29
30         public Controller()
31         {
32             GetDatas();
33         }
34
35         /// <summary>
36         /// Get All datas and links all objects between them
37         /// </summary>
38         /// <returns>a list of project containing all infos</returns>
39         private void GetDatas()
40         {
41             //Initialise variables
42             List<Project> allProjects;
43             List<UserStory> allUserStories;
44             List<Sprint> allSprints;
45             List<State> allStates;
46             List<User> allUsers;
47             List<Classes.File> allFiles;
48             List<Activity> allActivities;
49             List<Checklist> allChecklists;
50             List<ChecklistItem> allChecklistItems;
51             List<Priority> allPriorities;
52             List<Comment> allComments;
53             List<Classes.Type> allTypes;
54             List<Node> allNodes;
55             List<MindMap> allMindMaps;
56
57             List<int>[] projectStates;
58             List<int>[] userStoriesSprint;
59             List<int>[] userUserStories;
60             List<int>[] userChecklistItem;
61             List<int>[] userProjects;
62
63             allProjects = DB.GetProjects();
64             allSprints = DB.GetSprints();
65             allUserStories = DB.GetUserStories();
66             allStates = DB.GetStates();
67             allUsers = DB.GetUsers();
68             allFiles = DB.GetFiles();
69             allActivities = DB.GetActivities();
70             allChecklists = DB.GetChecklists();
71             allChecklistItems = DB.GetChecklistItems();
72             allPriorities = DB.GetPriorities();
73             allComments = DB.GetComments();
```

```

74    allTypes = DB.GetTypes();
75    allNodes = DB.GetNodes();
76    allMindMaps = DB.GetMindMaps();
77
78
79    #region Link State with project
80    projectStates = DB.GetProjectStates();
81    // 0:IdProject,1:IdState,2:order
82    foreach (int[] item in projectStates)
83    {
84        List<Project> project = allProjects.Where(p => p.Id == ←
85            item[0]).ToList();
86        List<State> state = allStates.Where(s => s.Id == ←
87            item[1]).ToList();
88
89        if (project.Count == 1 && state.Count == 1)
90        {
91            project[0].States.Add(item[2], state[0]);
92        }
93    }
94    #endregion
95    #region Link UserStory with other classes
96    foreach (UserStory u in allUserStories)
97    {
98        List<Project> project = allProjects.Where(p => p.Id == ←
99            u.ProjectId).ToList();
100       List<State> state = allStates.Where(s => s.Id == ←
101          u.StateId).ToList();
102       List<Priority> priority = allPriorities.Where(p => p.Id == ←
103          u.PriorityId).ToList();
104       List<Classes.Type> type = allTypes.Where(t => t.Id == ←
105          u.TypeId).ToList();
106
106       List<Activity> activities = allActivities.Where(a => ←
107           a.UserStoryId == u.Id).ToList();
108       List<Classes.File> files = allFiles.Where(f => f.UserStoryId == ←
109           u.Id).ToList();
110       List<Checklist> checklists = allChecklists.Where(c => ←
111           c.UserStoryId == u.Id).ToList();
112       List<Comment> comments = allComments.Where(c => c.UserStoryId ←
113           == u.Id).ToList();
114
114       if (state.Count == 1)
115       {
116           u.State = state[0];
117       }
118       if (priority.Count == 1)
119       {
120           u.Priority = priority[0];
121       }
122       if (type.Count == 1)
123       {
124           u.Type = type[0];
125       }
126       if (project.Count == 1)
127       {
128           project[0].AllUserStories.Add(u);
129           u.Project = project[0];
130       }
131       u.Activities = activities;
132       foreach (Activity a in activities)
133       {
134           a.UserStory = u;
135       }
136       u.Checklists = checklists;
137       foreach (Checklist c in checklists)
138       {
139           c.UserStory = u;
140       }
141       u.Comments = comments;
142       foreach (Comment c in comments)
143       {
144           List<User> users = allUsers.Where(user => user.Id == ←
145               c.UserId).ToList();
146           if (users.Count == 1)
147           {
148               c.User = users[0];
149               c.UserStory = u;
150           }
151       }
152   }
153 }
```

```

142         }
143     }
144
145     u.Files = files;
146     foreach (Classes.File f in files)
147     {
148         f.UserStory = u;
149     }
150 }
151 #endregion
152 #region Link UserStory with Sprint
153 userStoriesSprint = DB.GetUserStoriesSprint();
154 // 0: IdUserStory ,1: IdSprint ,2: Order
155 foreach (int[] item in userStoriesSprint)
156 {
157     List<UserStory> userStory = allUserStories.Where(u => u.Id == ←
158         item[0]).ToList();
159     List<Sprint> sprint = allSprints.Where(s => s.Id == ←
160         item[1]).ToList();
161     if (userStory.Count == 1 && sprint.Count == 1)
162     {
163         sprint[0].AddUserStory(item[2], userStory[0]);
164     }
165 }
166 #endregion
167 #region Link Sprint with Project
168 foreach (Sprint s in allSprints)
169 {
170     List<Project> project = allProjects.Where(p => p.Id == ←
171         s.ProjectId).ToList();
172     if (project.Count == 1)
173     {
174         s.Project = project[0];
175         project[0].Sprints.Add(s);
176     }
177 }
178 #endregion
179 #region Link Checklist with ChecklistItems
180 foreach (Checklist chk in allChecklists)
181 {
182     List<ChecklistItem> checklistItems = allChecklistItems.Where(c ←
183         => c.ChecklistId == chk.Id).ToList();
184     chk.ChecklistItems = checklistItems;
185     foreach (ChecklistItem chkItm in checklistItems)
186     {
187         chkItm.Checklist = chk;
188     }
189 }
190 #endregion
191 #region Link user with UserStory
192 userUserStories = DB.GetUserUserStory();
193 foreach (int[] item in userUserStories)
194 {
195     // 0: IdUser ,1: IdUserStories
196     List<User> user = allUsers.Where(u => u.Id == item[0]).ToList();
197     List<UserStory> userStory = allUserStories.Where(us => us.Id == ←
198         item[1]).ToList();
199     if (userStory.Count == 1 && user.Count == 1)
200     {
201         userStory[0].AddUser(user[0]);
202     }
203 }
204 #endregion
205 #region Link user with CheckListItem
206 userChecklistItem = DB.GetUserChecklistItem();
207 foreach (int[] item in userChecklistItem)
208 {
209     // 0: IdUser ,1: IdChecklistItem
210     List<User> user = allUsers.Where(u => u.Id == item[0]).ToList();
211     List<ChecklistItem> checklistItem = allChecklistItems.Where(c ←
212         => c.Id == item[1]).ToList();
213     if (checklistItem.Count == 1 && user.Count == 1)
214     {
215         checklistItem[0].AddUser(user[0]);
216     }
217 }
218 #endregion
219 #region Link user with Project

```

```

215     userProjects = DB.GetUserProject();
216     foreach (int[] item in userProjects)
217     {
218         // 0: IdUser ,1: IdProject
219         List<User> user = allUsers.Where(u => u.Id == item[0]).ToList();
220         List<Project> project = allProjects.Where(p => p.Id == ←
221             item[1]).ToList();
222         if (project.Count == 1 && user.Count == 1)
223         {
224             project[0].AddUser(user[0]);
225         }
226     }
227 #endregion
228 #region Link Node with Nodes and mindmaps
229 foreach (Node node in allNodes)
230 {
231     List<MindMap> mindMaps = allMindMaps.Where(m => m.Id == ←
232         node.MindmapId).ToList();
233     if (mindMaps.Count == 1)
234     {
235         node.MindMap = mindMaps[0];
236     }
237     //verify if root node or not
238     if (node.PreviousId == null)
239     {
240         if (mindMaps.Count == 1)
241         {
242             mindMaps[0].Root = node;
243         }
244     }
245     else
246     {
247         List<Node> nodes = allNodes.Where(n => n.Id == ←
248             node.PreviousId).ToList();
249         if (nodes.Count == 1)
250         {
251             node.Previous = nodes[0];
252         }
253     }
254     List<Node> childrens = allNodes.Where(n => n.PreviousId == ←
255         node.Id).ToList();
256     node.Childrens = childrens;
257 }
258 #endregion
259 #region Link MindMaps with projects
260 foreach (MindMap mindMap in allMindMaps)
261 {
262     List<Project> projects = allProjects.Where(p => p.Id == ←
263         mindMap.ProjectId).ToList();
264     if (projects.Count == 1)
265     {
266         mindMap.Project = projects[0];
267         projects[0].MindMaps.Add(mindMap);
268     }
269 }
270 #endregion
271
272 projects = allProjects;
273 users = allUsers;
274 types = allTypes;
275 priorities = allPriorities;
276 states = allStates;
277
278
279 #region Remove Methods
280 //All the creations methods remove the object in controller's values ←
281 //and tell DB to remove them
282 public bool Delete(Project project)
283 {
284     bool result = DB.Delete(project);
285     projects.Remove(project);
286     return result;
287 }
288 public bool Delete(Checklist checklist)

```

```

288     {
289         bool result = DB.Delete(checklist);
290         checklist.UserStory.Checklists.Remove(checklist);
291         return result;
292     }
293     public bool Delete(User user)
294     {
295         bool result = DB.Delete(user);
296         users.Remove(user);
297         return result;
298     }
299     public bool Delete(State state)
300     {
301         List<Project> removingState = projects.Where(p => ←
302             p.States.ContainsValue(state)).ToList();
303         foreach (Project project in removingState)
304         {
305             RemoveStateFromProject(state, project);
306         }
307         this.States.Remove(state);
308         bool result = DB.Delete(state);
309         return result;
310     }
311     public bool Delete(Sprint sprint)
312     {
313         sprint.Project.Sprints.Remove(sprint);
314         bool result = DB.Delete(sprint);
315         return result;
316     }
317     public bool Delete(ChecklistItem checklistItem)
318     {
319         checklistItem.Checklist.ChecklistItems.Remove(checklistItem);
320         bool result = DB.Delete(checklistItem);
321         return result;
322     }
323     public bool Delete(UserStory userStory)
324     {
325         List<Sprint> sprintRemoveUserStory = ←
326             userStory.Project.Sprints.Where(s => ←
327                 s.OrderedUserStories.ContainsValue(userStory)).ToList();
328         foreach (Sprint sprint in sprintRemoveUserStory)
329         {
330             RemoveUserStoryFromSprint(userStory, sprint);
331         }
332         userStory.Project.AllUserStories.Remove(userStory);
333         bool result = DB.Delete(userStory);
334         return result;
335     }
336     public bool Delete(Comment comment)
337     {
338         bool result = DB.Delete(comment);
339         return result;
340     }
341     public bool Delete(Activity activity)
342     {
343         bool result = DB.Delete(activity);
344         return result;
345     }
346     public bool Delete(Classes.File file)
347     {
348         bool result = DB.Delete(file);
349         file.UserStory.Files.Remove(file);
350         return result;
351     }
352     public bool RemoveStateFromProject(State state, Project project)
353     {
354         int order = -1;
355         if (project.States.ContainsValue(state) && project.States.Count > 1)
356         {
357             foreach (KeyValuePair<int, State> item in project.States)
358             {
359                 if (item.Value == state)
360                 {
361                     project.States.Remove(item.Key);
362                     order = item.Key;
363
364                     foreach (UserStory userStory in ←
365                         project.AllUserStories.Where(u => u.State.Id == ←
366                             state.Id))

```

```

362             {
363                 UserStorySwitchState(userStory, ←
364                     project.States.First().Value);
365             }
366             break;
367         }
368     }
369     DB.RemoveStateFromProject(project, order);
370 }
371 else
372 {
373     return false;
374 }
375 return true;
376 }
377 public bool RemoveUserFromIUsersAssigned(User user, IUsersAssigned ←
378     usersAssigned)
379 {
380     string typeName = usersAssigned.GetType().Name;
381     switch (typeName)
382     {
383         case "Project":
384             RemoveUserFromProject(user, usersAssigned as Project);
385             break;
386         case "UserStory":
387             RemoveUserFromUserStory(user, usersAssigned as UserStory);
388             CreateActivity(string.Format("{0} \u2192 t \u21d3 sassign ", ←
389                             user), usersAssigned as UserStory);
390             break;
391         case "ChecklistItem":
392             RemoveUserFromChecklistItem(user, usersAssigned as ←
393                             ChecklistItem);
394             break;
395         default:
396             return false;
397     }
398     return true;
399 }
400 private bool RemoveUserFromChecklistItem(User user, ChecklistItem ←
401     checklistItem)
402 {
403     bool result = DB.RemoveUserFromChecklistItem(user, checklistItem);
404     checklistItem.RemoveUser(user);
405     return result;
406 }
407 private bool RemoveUserFromUserStory(User user, UserStory userStory)
408 {
409     bool result = DB.RemoveUserFromUserStory(user, userStory);
410     CreateActivity(string.Format("\\"{0}\\ \u2192 t \u21d3 sassign ", user), ←
411                     userStory);
412     userStory.RemoveUser(user);
413     return result;
414 }
415 private bool RemoveUserFromProject(User user, Project project)
416 {
417     bool result = DB.RemoveUserFromProject(user, project);
418     project.RemoveUser(user);
419     return result;
420 }
421 public bool RemoveUserStoryFromSprint(UserStory userStory, Sprint sprint)
422 {
423     bool result = false;
424     if (sprint.OrderedUserStories.ContainsValue(userStory))
425     {
426         int order = 0;
427         while (!sprint.OrderedUserStories.ContainsKey(order) || ←
428             sprint.OrderedUserStories[order] != userStory)
429         {
430             order++;
431         }
432         CreateActivity(string.Format("A \u2192 t \u21d3 sassign \u21d3 du \u2192 sprint \u21d3 \u2192 \\"{0}\\\"", sprint), userStory);
433         sprint.RemoveUserStoryByOrder(order);
434         result = DB.RemoveUserStoryFromSprint(userStory, sprint, order);
435     }
436     return result;
437 }
438 public bool Delete(MindMap mindMap)

```

```

433     {
434         bool result = DB.Delete(mindMap);
435         mindMap.Project.MindMaps.Remove(mindMap);
436         return result;
437     }
438     public bool Delete(Node node)
439     {
440         bool result = DB.Delete(node);
441         if (!result)
442         {
443             return result;
444         }
445         foreach (Node n in node.Childrens)
446         {
447             n.Previous = node.Previous;
448             node.Previous.Childrens.Add(n);
449         }
450         node.Previous.Childrens.Remove(node);
451         return result;
452     }
453
454 #endregion
455
456 #region Update Methods
457 //All the creations methods update the objects in controller's values ←
458 //and send them to DB
459 public bool UserStorySwitchState(UserStory userStory, State state)
{
460     bool result = false;
461     if (userStory.State != state)
462     {
463         CreateActivity(string.Format("Pass ↴de ↴l' tat ↴\"{0}\" ↴ u←
464             l' tat ↴\"{1}\"", userStory.State, state), userStory);
465         result = DB.UpdateUserStory(userStory.Description, ←
466             userStory.DateLimit, userStory.ComplexityEstimation, ←
467             userStory.CompletedComplexity, userStory.Blocked, ←
468             userStory.Priority, state, userStory.Type, userStory);
469         userStory.State = state;
470     }
471     return result;
472 }
473 public bool UpdateCheckListItem(string nameItem, bool done, ←
474     ChecklistItem item)
{
475     bool result = DB.UpdateCheckListItem(nameItem, done, item);
476     item.NameItem = nameItem;
477     item.Done = done;
478     return result;
479 }
480 public bool UpdateUserStory(string description, DateTime? selectedDate, ←
481     int complexity, int completedComplexity, bool blocked, Priority ←
482     aPriority, Classes.Type aType, State aState, UserStory userStory)
{
483     bool result = false;
484     //Verify if anything is different to create an activity consequently
485     if (userStory.Description != description ||
486         userStory.DateLimit != selectedDate ||
487         userStory.ComplexityEstimation != complexity ||
488         userStory.CompletedComplexity != completedComplexity ||
489         userStory.Blocked != blocked ||
490         userStory.Priority != aPriority ||
491         userStory.Type != aType ||
492         userStory.State != aState)
493     {
494         CreateActivity("Les ↴informations ↴ont ↴ t ↴ umises ↴ ujour", ←
495             userStory);
496         result = DB.UpdateUserStory(description, selectedDate, ←
497             complexity, completedComplexity, blocked, aPriority, aState, ←
498             aType, userStory);
499         userStory.Description = description;
500         userStory.DateLimit = selectedDate;
501         userStory.ComplexityEstimation = complexity;
502         userStory.CompletedComplexity = completedComplexity;
503         userStory.Blocked = blocked;
504         userStory.State = aState;
505         userStory.Priority = aPriority;
506         userStory.Type = aType;
507     }
508     return result;

```

```

501     }
502     public bool UpdateFile(string fileDescription, Classes.File file)
503     {
504         bool result = DB.UpdateFile(fileDescription, file);
505         file.Description = fileDescription;
506         return result;
507     }
508     public bool UpdateCheckList(string name, List<ChecklistItem> items, ←
509     Checklist checklist)
510     {
511         bool result = DB.UpdateCheckList(name, checklist);
512         checklist.Name = name;
513         checklist.ChecklistItems.Clear();
514         foreach (ChecklistItem item in items)
515         {
516             if (item.Id == -1)
517             {
518                 checklist.ChecklistItems.Add(DB.CreateCheckListItem(item.NameItem, ←
519                     checklist));
520             }
521             else
522             {
523                 UpdateCheckListItem(item.NameItem, item.Done, item);
524                 checklist.ChecklistItems.Add(item);
525             }
526         }
527         return result;
528     }
529     public bool UpdateProject(string name, string description, DateTime ←
530     date, Project aProject)
531     {
532         bool result = DB.UpdateProject(name, description, date, aProject);
533         aProject.Name = name;
534         aProject.Description = description;
535         aProject.Begin = date;
536         return result;
537     }
538     public bool UpdateState(string text, State state)
539     {
540         bool result = DB.UpdateState(text, state);
541         state.Name = text;
542         return result;
543     }
544     public bool UpdateSprint(DateTime begin, DateTime end, Sprint sprint)
545     {
546         bool result = DB.UpdateSprint(begin, end, sprint);
547         sprint.Begin = begin;
548         sprint.End = end;
549         return result;
550     }
551     public bool UpdateUser(string text, User user)
552     {
553         bool result = DB.UpdateUser(text, user);
554         user.Name = text;
555         return result;
556     }
557     public bool UpdateMindMap(string text, MindMap mindMap)
558     {
559         //Update root node to have same text
560         bool result = DB.UpdateMindMap(text, mindMap);
561         mindMap.Name = text;
562         result = result && UpdateNode(text, null, mindMap.Root);
563         return result;
564     }
565     public bool UpdateNode(string text, Node previous, Node node)
566     {
567         bool result = DB.UpdateNode(text, previous, node);
568         if (!result)
569         {
570             return result;
571         }
572         if (node.Previous != previous)
573         {
574             //if current previous is a children, redirect to current parent
575             if (node.AllChildrens().Contains(previous))
576             {
577                 List<Node> directChilds = node.Childrens;
578                 for (int i = 0; i < directChilds.Count; i++)
579                 {
580                     if (directChilds[i].Previous == previous)
581                     {
582                         directChilds[i].Previous = node;
583                     }
584                 }
585             }
586         }
587     }

```

```

577             {
578                 Node n = directChilds[i];
579                 UpdateNode(n.Name, node.Previous, n);
580             }
581         }
582         //assign new values
583         node.Previous.Childrens.Remove(node);
584         node.Previous = previous;
585         previous.Childrens.Add(node);
586     }
587     node.Name = text;
588
589     return result;
590 }
591 #endregion
592
593 #region Creation Methods
594 //All the creations methods add the objects in controller's values and ←
595 //send them to DB
596 public bool AddUserStoryToSprint(UserStory userStory, Sprint sprint)
597 {
598     //Create sprint at the next order, create an activity and return ←
599     //success
600     if (!sprint.OrderedUserStories.ContainsValue(userStory))
601     {
602         int order = 0;
603         while (sprint.OrderedUserStories.ContainsKey(order))
604         {
605             order++;
606         }
607         sprint.AddUserStory(order, userStory);
608         DB.AddUserStoryToSprint(userStory, sprint, order);
609         CreateActivity(string.Format("At uajout uausprint←
610             \"{0}\"", sprint), userStory);
611         return true;
612     }
613     else
614     {
615         return false;
616     }
617 }
618 public bool AddStateToProject(State state, Project project)
619 {
620     //add state to next after creating and checking
621     int i = 0;
622     while (project.States.ContainsKey(i))
623     {
624         i++;
625     }
626     bool result = DB.AddStateToProject(state, project, i);
627     project.States.Add(i, state);
628     return result;
629 }
630 public bool AddUserToIUsersAssigned(User user, IUsersAssigned ←
631 usersAssigned)
632 {
633     //Verify which IUsersAssigned it is
634     string typeName = usersAssigned.GetType().Name;
635     switch (typeName)
636     {
637         case "Project":
638             AddUserToProject(user, usersAssigned as Project);
639             break;
640         case "UserStory":
641             AddUserToUserStory(user, usersAssigned as UserStory);
642             break;
643         case "ChecklistItem":
644             AddUserToChecklistItem(user, usersAssigned as ChecklistItem);
645             break;
646         default:
647             return false;
648     }
649     return true;
650 }
651 private bool AddUserToUserStory(User user, UserStory userStory)
652 {
653     //create, check, create activity, assign values and return
654     bool result = DB.AddUserToUserStory(user, userStory);
655 }
```

```

651     CreateActivity(string.Format("\'{0}\' au t assign ", user), ←
652         userStory);
653     userStory.AddUser(user);
654     return result;
655 }
656 private bool AddUserToChecklistItem(User user, ChecklistItem ←
657     checklistItem)
658 {
659     //create, check, assign values and return
660     bool result = DB.AddUserToChecklistItem(user, checklistItem);
661     checklistItem.AddUser(user);
662     return result;
663 }
664 private bool AddUserToProject(User user, Project project)
665 {
666     //create, check, assign values and return
667     bool result = DB.AddUserToProject(user, project);
668     project.AddUser(user);
669     return result;
670 }
671 public bool CreateActivity(string description, UserStory userStory)
672 {
673     //create, check, assign values and return
674     Activity activity = DB.CreateActivity(description, DateTime.Now, ←
675         userStory);
676     bool result = activity != null;
677     userStory.Activities.Add(activity);
678     activity.UserStory = userStory;
679     return result;
680 }
681 public bool CreateUserStory(string description, DateTime? selectedDate, ←
682     int complexity, Priority aPriority, Classes.Type aType, Project ←
683     aProject)
684 {
685     //create and check
686     UserStory userStory = DB.CreateUserStory(description, selectedDate, ←
687         complexity, aPriority, aType, aProject.States.First().Value, ←
688         aProject);
689     bool result = userStory != null;
690
691     //assign values, create activity and return
692     userStory.Priority = aPriority;
693     userStory.Type = aType;
694     userStory.State = aProject.States.First().Value;
695     userStory.Project = aProject;
696     aProject.AllUserStories.Add(userStory);
697     CreateActivity("Au t cr ", userStory);
698     return result;
699 }
700 public bool CreateSprint(DateTime dateBegin, DateTime dateEnd, Project ←
701     aProject)
702 {
703     //create, check, assign values and return
704     Sprint sprint = DB.CreateSprint(dateBegin, dateEnd, aProject);
705     bool result = sprint != null;
706     sprint.Project = aProject;
707     aProject.Sprints.Add(sprint);
708     return result;
709 }
710 public bool CreateFile(string fileName, string description, UserStory ←
711     userStory)
712 {
713     //create and check
714     Classes.File file = DB.CreateFile(fileName, description, userStory);
715     bool result = file != null;
716
717     //assign values, create activity and return
718     userStory.Files.Add(file);
719     file.UserStory = userStory;
720     CreateActivity(string.Format("\'{0}\' au t uli ", file), ←
721         userStory);
722     return result;
723 }
724 public bool CreateUser(string name, IUsersAssigned usersAssigned)
725 {
726     //create and check
727     User user = DB.CreateUser(name);
728     bool result = user != null;

```

```

720 //Add user to parent to be viewed in userMenu directly
721 string typeName = usersAssigned.GetType().Name;
722 switch (typeName)
723 {
724     case "UserStory":
725         AddUserToProject(user, (usersAssigned as UserStory).Project);
726         break;
727     case "ChecklistItem":
728         AddUserToUserStory(user, (usersAssigned as ChecklistItem).Checklist.UserStory);
729         break;
730     case "Project":
731     default:
732         break;
733 }
734
735 //assign values and return
736 users.Add(user);
737 return result;
738 }
739 public bool CreateState(string name)
740 {
741     //create,check, assign values and return
742     State state = DB.CreateState(name);
743     bool result = state != null;
744     states.Add(state);
745     return result;
746 }
747 public bool CreateComment(string text, User user, UserStory userStory)
748 {
749     //create and verify
750     Comment comment = DB.CreateComment(text, userStory, user);
751     bool result = comment != null;
752
753     //assign values, create activity and return
754     comment.UserStory = userStory;
755     userStory.Comments.Add(comment);
756     comment.User = user;
757
758     CreateActivity(string.Format("\u2022 {0} \u2022 comment ", user), userStory);
759     return result;
760 }
761 public bool CreateCheckList(string aName, Checklist checklist)
762 {
763     //create and verify
764     ChecklistItem checklistItem = DB.CreateCheckList(aName, checklist);
765     bool result = checklistItem != null;
766
767     //assign values and return
768     checklistItem.Checklist = checklist;
769     checklist.ChecklistItems.Add(checklistItem);
770     return result;
771 }
772 public Checklist CreateCheckList(string aName, UserStory aUserStory)
773 {
774     //Create, assign values and create activity
775     Checklist checklist = DB.CreateCheckList(aName, aUserStory);
776     aUserStory.Checklists.Add(checklist);
777     checklist.UserStory = aUserStory;
778     CreateActivity(string.Format("La\u2022 checklist \u2022 {0} \u2022 t \u2022 cr \u2022 e ", checklist), aUserStory);
779     return checklist;
780 }
781 public bool CreateProject(string aName, string aDesc, DateTime aDate)
782 {
783     //Create and determine if created correctly
784     Project project = DB.CreateProject(aName, aDesc, aDate);
785     bool result = project != null;
786     projects.Add(project);
787
788     //Assign 3 states by default and create if necessary
789     if (states.Count < 3)
790     {
791         CreateState("A\u2022faire");
792         CreateState("En\u2022cours");
793         CreateState("Accompli");
794     }
795     project.States.Add(0, states[0]);

```

```

796     project.States.Add(1, states[1]);
797     project.States.Add(2, states[2]);
798     DB.AddStateToProject(states[0], project, 0);
799     DB.AddStateToProject(states[1], project, 1);
800     DB.AddStateToProject(states[2], project, 2);
801 
802     return result;
803 }
804 public bool CreateMindMap(string aName, Project project)
805 {
806     //Create and verify
807     MindMap mindMap = DB.CreateMindmap(aName, project);
808     bool result = mindMap != null;
809     //assign values and return
810     mindMap.Project = project;
811     mindMap.Root = DB.CreateNode(aName, null, mindMap);
812     project.MindMaps.Add(mindMap);
813     return result;
814 }
815 public bool CreateNode(string aName, Node previous, MindMap mindMap)
816 {
817     //Create and verify
818     Node node = DB.CreateNode(aName, previous, mindMap);
819     bool result = node != null;
820 
821     //set root if previous null and assign values
822     if (previous != null)
823     {
824         previous.Childrens.Add(node);
825     }
826     else
827     {
828         mindMap.Root = node;
829     }
830     node.Previous = previous;
831     node.MindMap = mindMap;
832     return result;
833 }
834 #endregion
835
836 }
837

```

Listing 83 – ../../Scrum'o'wall/Scrum'o'wall/Controls/Controller.cs

4.2 DB.cs

```

1 /*
2 * Author : Gal Serge Mariot
3 * Project : Scrum'o'wall
4 * File : DB.cs
5 * Desc. : This file is a class to access to the database
6 */
7 using Microsoft.Win32;
8 using Scrum_o_wall.Classes;
9 using System;
10 using System.Collections.Generic;
11 using System.Data.OleDb;
12 using System.IO;
13 using System.Windows;
14
15 namespace Scrum_o_wall
16 {
17     public static class DB
18     {
19         public static string DbFileName;
20         private static OleDbConnection connection;
21         private static OleDbConnection GetConnection()
22         {
23             if (connection is null && DbFileName == null)
24             {
25                 OpenFileDialog opf = new OpenFileDialog
26                 {
27                     Title = "Quel base de données utiliser ?",
28                     DefaultExt = ".accdb|*.accdb",
29                     Filter = "Fichier Accès (*.accdb)|*.accdb"
30                 };

```

```

31         do
32         {
33             if (opf.ShowDialog() != true)
34             {
35                 Application.Current.Shutdown();
36             }
37         } while (!opf.SafeFileName.Contains(".accdb"));
38         connection = new ←
39             OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data←
40             Source=" + opf.FileName + ";PersistSecurityInfo=False;");
41     }
42     else if (connection is null)
43     {
44         connection = new ←
45             OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data←
46             Source=" + DbFileName + ";PersistSecurityInfo=False;");
47     }
48
49     return connection;
50 }
51
52
53 #region UPDATE
54 //Update the objects and send true if exactly one line has changed
55 public static bool UpdateUserStory(string description, DateTime? ←
56     selectedDate, int complexity, int completedComplexity, bool blocked, ←
57     Priority priority, State state, Classes.Type type, UserStory userStory)
58 {
59     //Initialize variables
60     OleDbCommand cmd;
61     bool result;
62
63     //Open database, build sql statement and prepare
64     DB.GetConnection().Open();
65     cmd = DB.GetConnection().CreateCommand();
66     cmd.CommandText = "UPDATE [UserStories] SET [DescriptionUserStory]=?←
67         , [DateLimite]=?, [ComplexityEstimation]=?, [CompletedComplexity]=?←
68         , [Blocked]=?, [IdPriority]=?, [IdState]=?, [IdType]=? WHERE [←
69         IdUserStory]=?";
70     cmd.Parameters.Add("DescriptionUserStory", OleDbType.LongVarChar, ←
71         65535);
72     cmd.Parameters.Add("DateLimite", OleDbType.Date);
73     cmd.Parameters.Add("ComplexityEstimation", OleDbType.Integer);
74     cmd.Parameters.Add("CompletedComplexity", OleDbType.Integer);
75     cmd.Parameters.Add("Blocked", OleDbType.Boolean);
76     cmd.Parameters.Add("IdPriority", OleDbType.Integer);
77     cmd.Parameters.Add("IdState", OleDbType.Integer);
78     cmd.Parameters.Add("IdType", OleDbType.Integer);
79     cmd.Parameters.Add("IdUserStory", OleDbType.Integer);
80     cmd.Parameters[0].Value = description;
81     if (selectedDate == null)
82     {
83         cmd.Parameters[1].Value = DBNull.Value;
84     }
85     else
86     {
87         cmd.Parameters[1].Value = selectedDate;
88     }
89     cmd.Parameters[2].Value = complexity;
90     cmd.Parameters[3].Value = completedComplexity;
91     cmd.Parameters[4].Value = blocked;
92     cmd.Parameters[5].Value = priority.Id;
93     cmd.Parameters[6].Value = state.Id;
94     cmd.Parameters[7].Value = type.Id;
95     cmd.Parameters[8].Value = userStory.Id;
96
97     //Execute sql statement
98     cmd.Prepare();
99     result = cmd.ExecuteNonQuery() == 1;

```

```

100    bool result;
101
102    //Open database, build sql statement and prepare
103    DB.GetConnection().Open();
104    cmd = DB.GetConnection().CreateCommand();
105    cmd.CommandText = "UPDATE [TFiles] SET [DescriptionFile] = ? WHERE [←
106        IdFile] = ?";
107    cmd.Parameters.Add("DescriptionFile", OleDbType.LongVarChar, 65535);
108    cmd.Parameters.Add("IdFile", OleDbType.Integer);
109    cmd.Parameters[0].Value = fileDescription;
110    cmd.Parameters[1].Value = file.Id;
111
112    //Execute sql statement
113    cmd.Prepare();
114    result = cmd.ExecuteNonQuery() == 1;
115
116    //Close database
117    DB.GetConnection().Close();
118    return result;
119}
120public static bool UpdateCheckList(string name, Checklist checklist)
121{
122    //Initialize variables
123    OleDbCommand cmd;
124    bool result;
125
126    //Open database, build sql statement and prepare
127    DB.GetConnection().Open();
128    cmd = DB.GetConnection().CreateCommand();
129    cmd.CommandText = "UPDATE [TChecklists] SET [NameChecklist] = ? WHERE [←
130        IdChecklist] = ?";
131    cmd.Parameters.Add("NameChecklist", OleDbType.VarChar, 255);
132    cmd.Parameters.Add("IdChecklist", OleDbType.Integer);
133    cmd.Parameters[0].Value = name;
134    cmd.Parameters[1].Value = checklist.Id;
135
136    //Execute sql statement
137    cmd.Prepare();
138    result = cmd.ExecuteNonQuery() == 1;
139
140    //Close database
141    DB.GetConnection().Close();
142    return result;
143}
144public static bool UpdateCheckListItem(string nameItem, bool done, ←
145    ChecklistItem checklistItem)
146{
147    //Initialize variables
148    OleDbCommand cmd;
149    bool result;
150
151    //Open database, build sql statement and prepare
152    DB.GetConnection().Open();
153    cmd = DB.GetConnection().CreateCommand();
154    cmd.CommandText = "UPDATE [TChecklistItems] SET [NameItem] = ?, [Done] = ?←
155        WHERE [IdChecklistItem] = ?";
156    cmd.Parameters.Add("NameItem", OleDbType.VarChar, 255);
157    cmd.Parameters.Add("Done", OleDbType.Boolean);
158    cmd.Parameters.Add("IdChecklistItem", OleDbType.Integer);
159    cmd.Parameters[0].Value = nameItem;
160    cmd.Parameters[1].Value = done;
161    cmd.Parameters[2].Value = checklistItem.Id;
162
163    //Execute sql statement
164    cmd.Prepare();
165    result = cmd.ExecuteNonQuery() == 1;
166
167    //Close database
168    DB.GetConnection().Close();
169    return result;
170}
171public static bool UpdateProject(string name, string description, ←
172    DateTime dateTime, Project project)
173{
174    //Initialize variables

```

```

174     DB.GetConnection().Open();
175     cmd = DB.GetConnection().CreateCommand();
176     cmd.CommandText = "UPDATE TProjects SET NameProject=? , Description=? , DateBegin=? WHERE IdProject=? ;";
177     cmd.Parameters.Add("NameProject", OleDbType.VarChar, 50);
178     cmd.Parameters.Add("Description", OleDbType.LongVarChar, 65535);
179     cmd.Parameters.Add("DateBegin", OleDbType.Date);
180     cmd.Parameters.Add("IdProject", OleDbType.Integer);
181     cmd.Parameters[0].Value = name;
182     cmd.Parameters[1].Value = description;
183     cmd.Parameters[2].Value = dateDateTime;
184     cmd.Parameters[3].Value = project.Id;
185
186     //Execute sql statement
187     cmd.Prepare();
188     result = cmd.ExecuteNonQuery() == 1;
189
190     //Close database
191     DB.GetConnection().Close();
192     return result;
193 }
194 public static bool UpdateSprint(DateTime secBegin, DateTime secEnd, Sprint sprint)
195 {
196     //Initialize variables
197     OleDbCommand cmd;
198     bool result;
199
200     //Open database, build sql statement and prepare
201     DB.GetConnection().Open();
202     cmd = DB.GetConnection().CreateCommand();
203     cmd.CommandText = "UPDATE TSprints SET DateEnd=? , DateBegin=? WHERE IdSprint=? ;";
204     cmd.Parameters.Add("DateEnd", OleDbType.Date);
205     cmd.Parameters.Add("DateBegin", OleDbType.Date);
206     cmd.Parameters.Add("IdProject", OleDbType.Integer);
207     cmd.Parameters[0].Value = secEnd;
208     cmd.Parameters[1].Value = secBegin;
209     cmd.Parameters[2].Value = sprint.Id;
210
211     //Execute sql statement
212     cmd.Prepare();
213     result = cmd.ExecuteNonQuery() == 1;
214
215     //Close database
216     DB.GetConnection().Close();
217     return result;
218 }
219 public static bool UpdateUser(string text, User user)
220 {
221     //Initialize variables
222     OleDbCommand cmd;
223     bool result;
224
225     //Open database, build sql statement and prepare
226     DB.GetConnection().Open();
227     cmd = DB.GetConnection().CreateCommand();
228     cmd.CommandText = "UPDATE TUsers SET NameUser=? WHERE IdUser=? ;";
229     cmd.Parameters.Add("NameUser", OleDbType.Date);
230     cmd.Parameters.Add("IdUser", OleDbType.Integer);
231     cmd.Parameters[0].Value = text;
232     cmd.Parameters[1].Value = user.Id;
233
234     //Execute sql statement
235     cmd.Prepare();
236     result = cmd.ExecuteNonQuery() == 1;
237
238     //Close database
239     DB.GetConnection().Close();
240     return result;
241 }
242 public static bool UpdateState(string text, State state)
243 {
244     //Initialize variables
245     OleDbCommand cmd;
246     bool result;
247
248     //Open database, build sql statement and prepare
249     DB.GetConnection().Open();

```

```

250     cmd = DB.GetConnection().CreateCommand();
251     cmd.CommandText = "UPDATE TStates SET NameState=? WHERE IdState=?";
252     cmd.Parameters.Add("NameState", OleDbType.VarChar, 100);
253     cmd.Parameters.Add("IdState", OleDbType.Integer);
254     cmd.Parameters[0].Value = text;
255     cmd.Parameters[1].Value = state.Id;
256
257     //Execute sql statement
258     cmd.Prepare();
259     result = cmd.ExecuteNonQuery() == 1;
260
261     //Close database
262     DB.GetConnection().Close();
263     return result;
264 }
265 public static bool UpdateMindMap(string text, MindMap mindMap)
266 {
267     //Initialize variables
268     OleDbCommand cmd;
269     bool result;
270
271     //Open database, build sql statement and prepare
272     DB.GetConnection().Open();
273     cmd = DB.GetConnection().CreateCommand();
274     cmd.CommandText = "UPDATE TMindMaps SET NameMindMap=? WHERE IdMindMap=?";
275     cmd.Parameters.Add("NameMindMap", OleDbType.VarChar, 255);
276     cmd.Parameters.Add("IdMindMap", OleDbType.Integer);
277     cmd.Parameters[0].Value = text;
278     cmd.Parameters[1].Value = mindMap.Id;
279
280     //Execute sql statement
281     cmd.Prepare();
282     result = cmd.ExecuteNonQuery() == 1;
283
284     //Close database
285     DB.GetConnection().Close();
286     return result;
287 }
288 public static bool UpdateNode(string text, Node previous, Node node)
289 {
290     if ((previous == null && node.PreviousId != null) || (node.PreviousId == null && previous != null))
291     {
292         return false;
293     }
294     //Initialize variables
295     OleDbCommand cmd;
296     bool result;
297
298     //Open database, build sql statement and prepare
299     DB.GetConnection().Open();
300     cmd = DB.GetConnection().CreateCommand();
301     cmd.CommandText = "UPDATE TNodes SET NameNode=?, PreviousIdNode=? WHERE IdNode=?";
302     cmd.Parameters.Add("NameNode", OleDbType.VarChar, 255);
303     cmd.Parameters.Add("PreviousIdNode", OleDbType.Integer);
304     cmd.Parameters.Add("IdNode", OleDbType.Integer);
305     cmd.Parameters[0].Value = text;
306     if (previous == null)
307     {
308         cmd.Parameters[1].Value = DBNull.Value;
309     }
310     else
311     {
312         cmd.Parameters[1].Value = previous.Id;
313     }
314     cmd.Parameters[2].Value = node.Id;
315
316     //Execute sql statement
317     cmd.Prepare();
318     result = cmd.ExecuteNonQuery() == 1;
319
320     //Close database
321     DB.GetConnection().Close();
322     return result;
323 }
324

```

```

325     #endregion
326     #region ADD
327     //All the methods send back the objects after getting back the created ←
328     //    id (except linking methods)
329     public static Project CreateProject(string aName, string aDesc, ←
330     DateTime aDate)
331     {
332         //Initialize variables
333         OleDbCommand cmd;
334         int id;
335
336         //Open database, build sql statement and prepare
337         DB.GetConnection().Open();
338         cmd = DB.GetConnection().CreateCommand();
339         cmd.CommandText = "INSERT INTO TProjects ←
340             (NameProject, Description, DateBegin) VALUES (?, ?, ?);";
341         cmd.Parameters.Add("NameProject", OleDbType.VarChar, 50);
342         cmd.Parameters.Add("Description", OleDbType.LongVarChar, 65535);
343         cmd.Parameters.Add("DateBegin", OleDbType.Date);
344         cmd.Parameters[0].Value = aName;
345         cmd.Parameters[1].Value = aDesc;
346         cmd.Parameters[2].Value = aDate;
347
348         //Execute sql statement
349         cmd.Prepare();
350         cmd.ExecuteNonQuery();
351
352         //Get last inserted id
353         cmd.CommandText = "SELECT @@Identity;";
354         id = (int)cmd.ExecuteScalar();
355
356         //Close database
357         DB.GetConnection().Close();
358
359         //return created project
360         Project project = new Project(id, aName, aDesc, aDate);
361         return project;
362     }
363     public static ChecklistItem CreateCheckListItem(string aName, Checklist ←
364     checklist)
365     {
366         //Initialize variables
367         OleDbCommand cmd;
368         int id;
369
370         //Open database, build sql statement and prepare
371         DB.GetConnection().Open();
372         cmd = DB.GetConnection().CreateCommand();
373         cmd.CommandText = "INSERT INTO TChecklistItems ←
374             (NameItem, Done, IdChecklist) VALUES (?, ?, ?);";
375         cmd.Parameters.Add("NameItem", OleDbType.VarChar, 255);
376         cmd.Parameters.Add("Done", OleDbType.Boolean);
377         cmd.Parameters.Add("IdChecklist", OleDbType.Integer);
378         cmd.Parameters[0].Value = aName;
379         cmd.Parameters[1].Value = false;
380         cmd.Parameters[2].Value = checklist.Id;
381
382         //Execute sql statement
383         cmd.Prepare();
384         cmd.ExecuteNonQuery();
385
386         //Get last inserted id
387         cmd.CommandText = "SELECT @@Identity;";
388         id = (int)cmd.ExecuteScalar();
389
390         //Close database
391         DB.GetConnection().Close();
392
393         //return created project
394         ChecklistItem checklistItem = new ChecklistItem(id, aName, false, ←
395             checklist.Id);
396         return checklistItem;
397     }
398     public static Checklist CreateCheckList(string aName, UserStory userStory)
399     {
400         //Initialize variables
401         OleDbCommand cmd;
402         int id;

```

```

398     //Open database, build sql statement and prepare
399     DB.GetConnection().Open();
400     cmd = DB.GetConnection().CreateCommand();
401     cmd.CommandText = "INSERT INTO TChecklists ↵
402         (NameChecklist, IdUserStory) VALUES (?,?);";
403     cmd.Parameters.Add("NameChecklist", OleDbType.VarChar, 255);
404     cmd.Parameters.Add("IdUserStory", OleDbType.Integer);
405     cmd.Parameters[0].Value = aName;
406     cmd.Parameters[1].Value = userStory.Id;
407
408     //Execute sql statement
409     cmd.Prepare();
410     cmd.ExecuteNonQuery();
411
412     //Get last inserted id
413     cmd.CommandText = "SELECT @@Identity;";
414     id = (int)cmd.ExecuteScalar();
415
416     //Close database
417     DB.GetConnection().Close();
418
419     //return created project
420     Checklist checklist = new Checklist(id, aName, userStory.Id);
421     return checklist;
422 }
423 public static Activity CreateActivity(string description, DateTime now, ↵
424     UserStory userStory)
425 {
426     //Initialize variables
427     OleDbCommand cmd;
428     int id;
429
430     //Open database, build sql statement and prepare
431     DB.GetConnection().Open();
432     cmd = DB.GetConnection().CreateCommand();
433     cmd.CommandText = "INSERT INTO TAActivities ↵
434         (Description, DateTimeActivity, IdUserStory) VALUES (?, ?, ?);";
435     cmd.Parameters.Add("Description", OleDbType.LongVarChar, 65535);
436     cmd.Parameters.Add("DateTimeActivity", OleDbType.Date);
437     cmd.Parameters.Add("IdUserStory", OleDbType.Integer);
438     cmd.Parameters[0].Value = description;
439     cmd.Parameters[1].Value = now;
440     cmd.Parameters[2].Value = userStory.Id;
441
442     //Execute sql statement
443     cmd.Prepare();
444     cmd.ExecuteNonQuery();
445
446     //Get last inserted id
447     cmd.CommandText = "SELECT @@Identity;";
448     id = (int)cmd.ExecuteScalar();
449
450     //Close database
451     DB.GetConnection().Close();
452
453     //return created project
454     Activity activity = new Activity(id, description, now, userStory.Id);
455     return activity;
456 }
457 public static UserStory CreateUserStory(string description, DateTime? ↵
458     selectedDate, int complexity, Priority priority, Classes.Type type, ↵
459     State state, Project project)
460 {
461     //Initialize variables
462     OleDbCommand cmd;
463     int id;
464
465     //Open database, build sql statement and prepare
466     DB.GetConnection().Open();
467     cmd = DB.GetConnection().CreateCommand();
468     cmd.CommandText = "INSERT INTO TUserStories (DescriptionUserStory, ↵
469         DateLimite, ComplexityEstimation, CompletedComplexity, Blocked, ↵
470         IdProject, IdState, IdType, IdPriority) VALUES (?, ?, ?, ?, ?, ?, ↵
471         ?, ?, ?, ?);";
472     cmd.Parameters.Add("DescriptionUserStory", OleDbType.LongVarChar, ↵
473         65535);
474     cmd.Parameters.Add("DateLimite", OleDbType.Date);
475     cmd.Parameters.Add("ComplexityEstimation", OleDbType.Integer);
476     cmd.Parameters.Add("CompletedComplexity", OleDbType.Integer);

```

```

468     cmd.Parameters.Add("Blocked", OleDbType.Boolean);
469     cmd.Parameters.Add("IdProject", OleDbType.Integer);
470     cmd.Parameters.Add("IdState", OleDbType.Integer);
471     cmd.Parameters.Add("IdType", OleDbType.Integer);
472     cmd.Parameters.Add("IdPriority", OleDbType.Integer);
473     cmd.Parameters[0].Value = description;
474     if (selectedDate == null)
475     {
476         cmd.Parameters[1].Value = DBNull.Value;
477     }
478     else
479     {
480         cmd.Parameters[1].Value = selectedDate;
481     }
482     cmd.Parameters[2].Value = complexity;
483     cmd.Parameters[3].Value = 0;
484     cmd.Parameters[4].Value = false;
485     cmd.Parameters[5].Value = project.Id;
486     cmd.Parameters[6].Value = state.Id;
487     cmd.Parameters[7].Value = type.Id;
488     cmd.Parameters[8].Value = priority.Id;
489
490     //Execute sql statement
491     cmd.Prepare();
492     cmd.ExecuteNonQuery();
493
494     //Get last inserted id
495     cmd.CommandText = "SELECT @@Identity";
496     id = (int)cmd.ExecuteScalar();
497
498     //Close database
499     DB.GetConnection().Close();
500
501     //return created project
502     UserStory userStory = new UserStory(id, description, selectedDate, ←
503                                         complexity, 0, false, project.Id, state.Id, type.Id, priority.Id);
504     return userStory;
505 }
506 public static State CreateState(string name)
507 {
508     //Initialize variables
509     OleDbCommand cmd;
510     int id;
511
512     //Open database, build sql statement and prepare
513     DB.GetConnection().Open();
514     cmd = DB.GetConnection().CreateCommand();
515     cmd.CommandText = "INSERT INTO TStates(NameState) VALUES (?)";
516     cmd.Parameters.Add("NameState", OleDbType.VarChar, 30);
517     cmd.Parameters[0].Value = name;
518
519     //Execute sql statement
520     cmd.Prepare();
521     cmd.ExecuteNonQuery();
522
523     //Get last inserted id
524     cmd.CommandText = "SELECT @@Identity";
525     id = (int)cmd.ExecuteScalar();
526
527     //Close database
528     DB.GetConnection().Close();
529
530     //return created project
531     State state = new State(id, name);
532     return state;
533 }
534 public static Classes.File CreateFile(string fileName, string ←
535                                         description, UserStory userStory)
536 {
537     //Initialize variables
538     OleDbCommand cmd;
539     int id;
540
541     //Open database, build sql statement and prepare
542     DB.GetConnection().Open();
543     cmd = DB.GetConnection().CreateCommand();
544     cmd.CommandText = "INSERT INTO TFiles←
545                     (NameFile, DescriptionFile, IdUserStory) VALUES (?, ?, ?);";
546     cmd.Parameters.Add("NameState", OleDbType.LongVarChar, 65535);
547 }
```

```

544     cmd.Parameters.Add("DescriptionFile", OleDbType.LongVarChar, 65535);
545     cmd.Parameters.Add("IdUserStory", OleDbType.Integer);
546     cmd.Parameters[0].Value = fileName;
547     cmd.Parameters[1].Value = description;
548     cmd.Parameters[2].Value = userStory.Id;
549
550     //Execute sql statement
551     cmd.Prepare();
552     cmd.ExecuteNonQuery();
553
554     //Get last inserted id
555     cmd.CommandText = "SELECT @@Identity";
556     id = (int)cmd.ExecuteScalar();
557
558     //Close database
559     DB.GetConnection().Close();
560
561     //return created project
562     Classes.File file = new Classes.File(id, fileName, description, ↵
563                                         userStory.Id);
564     return file;
565 }
566 public static User CreateUser(string name)
567 {
568     //Initialize variables
569     OleDbCommand cmd;
570     int id;
571
572     //Open database, build sql statement and prepare
573     DB.GetConnection().Open();
574     cmd = DB.GetConnection().CreateCommand();
575     cmd.CommandText = "INSERT INTO TUsers (NameUser) VALUES (?)";
576     cmd.Parameters.Add("NameUser", OleDbType.VarChar, 255);
577     cmd.Parameters[0].Value = name;
578
579     //Execute sql statement
580     cmd.Prepare();
581     cmd.ExecuteNonQuery();
582
583     //Get last inserted id
584     cmd.CommandText = "SELECT @@Identity";
585     id = (int)cmd.ExecuteScalar();
586
587     //Close database
588     DB.GetConnection().Close();
589
590     //return created project
591     User user = new User(id, name);
592     return user;
593 }
594 public static Comment CreateComment(string name, UserStory userStory, ↵
595                                     User user)
596 {
597     //Initialize variables
598     OleDbCommand cmd;
599     int id;
600     DateTime dateTime = DateTime.Now;
601
602     //Open database, build sql statement and prepare
603     DB.GetConnection().Open();
604     cmd = DB.GetConnection().CreateCommand();
605     cmd.CommandText = "INSERT INTO TComments (DescriptionComment, CreationDateTime, IdUserStory, IdUser) VALUES (?,?,?,?,?)";
606     cmd.Parameters.Add("DescriptionComment", OleDbType.LongVarChar, ↵
607                        65535);
608     cmd.Parameters.Add("DateTime", OleDbType.Date);
609     cmd.Parameters.Add("IdUserStory", OleDbType.Integer);
610     cmd.Parameters.Add("IdUser", OleDbType.Integer);
611     cmd.Parameters[0].Value = name;
612     cmd.Parameters[1].Value = dateTime;
613     cmd.Parameters[2].Value = userStory.Id;
614     cmd.Parameters[3].Value = user.Id;
615
616     //Execute sql statement
617     cmd.Prepare();
618     cmd.ExecuteNonQuery();
619
620     //Get last inserted id

```

```

618     cmd.CommandText = "SELECT @@Identity";
619     id = (int)cmd.ExecuteScalar();
620
621     //Close database
622     DB.GetConnection().Close();
623
624     Comment comment = new Comment(id, name, dateTime, userStory.Id, ←
625         user.Id);
626     return comment;
627 }
628 public static Sprint CreateSprint(DateTime dateBegin, DateTime dateEnd, ←
629     Project project)
630 {
631     //Initialize variables
632     OleDbCommand cmd;
633     int id;
634
635     //Open database, build sql statement and prepare
636     DB.GetConnection().Open();
637     cmd = DB.GetConnection().CreateCommand();
638     cmd.CommandText = "INSERT INTO TSprints ←
639         (DateBegin, DateEnd, IdProject) VALUES (?, ?, ?);";
640     cmd.Parameters.Add("DateBegin", OleDbType.Date);
641     cmd.Parameters.Add("DateEnd", OleDbType.Date);
642     cmd.Parameters.Add("IdProject", OleDbType.Integer);
643     cmd.Parameters[0].Value = dateBegin;
644     cmd.Parameters[1].Value = dateEnd;
645     cmd.Parameters[2].Value = project.Id;
646
647     //Execute sql statement
648     cmd.Prepare();
649     cmd.ExecuteNonQuery();
650
651     //Get last inserted id
652     cmd.CommandText = "SELECT @@Identity";
653     id = (int)cmd.ExecuteScalar();
654
655     Sprint sprint = new Sprint(id, dateBegin, dateEnd, project.Id);
656     return sprint;
657 }
658 public static MindMap CreateMindmap(string name, Project project)
659 {
660     //Initialize variables
661     OleDbCommand cmd;
662     int id;
663
664     //Open database, build sql statement and prepare
665     DB.GetConnection().Open();
666     cmd = DB.GetConnection().CreateCommand();
667     cmd.CommandText = "INSERT INTO TMindMaps (NameMindMap, IdProject) ←
668         VALUES (?, ?);";
669     cmd.Parameters.Add("NameMindMap", OleDbType.VarChar, 255);
670     cmd.Parameters.Add("IdProject", OleDbType.Integer);
671     cmd.Parameters[0].Value = name;
672     cmd.Parameters[1].Value = project.Id;
673
674     //Execute sql statement
675     cmd.Prepare();
676     cmd.ExecuteNonQuery();
677
678     //Get last inserted id
679     cmd.CommandText = "SELECT @@Identity";
680     id = (int)cmd.ExecuteScalar();
681
682     //Close database
683     DB.GetConnection().Close();
684
685     MindMap mindMap = new MindMap(id, name, project.Id);
686     return mindMap;
687 }
688 public static Node CreateNode(string name, Node previous, MindMap mindMap)
689 {
690     //Initialize variables
691     OleDbCommand cmd;
692     int id;

```

```

693
694     //Open database, build sql statement and prepare
695     DB.GetConnection().Open();
696     cmd = DB.GetConnection().CreateCommand();
697     cmd.CommandText = "INSERT INTO TNodes ←
698         (NameNode, PreviousIdNode, IdMindMap) VALUES (?, ?, ?);";
699     cmd.Parameters.Add("NameNode", OleDbType.VarChar, 255);
700     cmd.Parameters.Add("PreviousIdNode", OleDbType.Integer);
701     cmd.Parameters.Add("IdMindMap", OleDbType.Integer);
702     cmd.Parameters[0].Value = name;
703     if (previous == null)
704     {
705         cmd.Parameters[1].Value = DBNull.Value;
706     }
707     else
708     {
709         cmd.Parameters[1].Value = previous.Id;
710     }
711     cmd.Parameters[2].Value = mindMap.Id;
712
713     //Execute sql statement
714     cmd.Prepare();
715     cmd.ExecuteNonQuery();
716
717     //Get last inserted id
718     cmd.CommandText = "SELECT @@Identity";
719     id = (int)cmd.ExecuteScalar();
720
721     //Close database
722     DB.GetConnection().Close();
723
724     int? previousId = null;
725     if (previous != null)
726     {
727         previousId = previous.Id;
728     }
729     Node node = new Node(id, name, previousId, mindMap.Id);
730     return node;
731 }
732 public static bool AddStateToProject(State state, Project project, int ←
733     order)
734 {
735     //Initialize variables
736     OleDbCommand cmd;
737     bool result;
738
739     //Open database, build sql statement and prepare
740     DB.GetConnection().Open();
741     cmd = DB.GetConnection().CreateCommand();
742     cmd.CommandText = "INSERT INTO TProjectStates ←
743         (IdProject, IdState, orderState) VALUES (?, ?, ?);";
744     cmd.Parameters.Add("IdProject", OleDbType.Integer);
745     cmd.Parameters.Add("IdState", OleDbType.Integer);
746     cmd.Parameters.Add("orderState", OleDbType.Integer);
747     cmd.Parameters[0].Value = project.Id;
748     cmd.Parameters[1].Value = state.Id;
749     cmd.Parameters[2].Value = order;
750
751     //Execute sql statement
752     cmd.Prepare();
753     result = cmd.ExecuteNonQuery() == 1;
754
755     //Close database
756     DB.GetConnection().Close();
757     return result;
758 }
759 public static bool AddUserToChecklistItem(User user, ChecklistItem ←
760     checklistItem)
761 {
762     //Initialize variables
763     OleDbCommand cmd;
764     bool result;
765
766     //Open database, build sql statement and prepare
767     DB.GetConnection().Open();
768     cmd = DB.GetConnection().CreateCommand();
769     cmd.CommandText = "INSERT INTO TUserChecklistItem ←
770         (IdUser, IdChecklistItem) VALUES (?, ?);";
771     cmd.Parameters.Add("IdUser", OleDbType.Integer);

```

```

767     cmd.Parameters.Add("IdProject", OleDbType.Integer);
768     cmd.Parameters[0].Value = user.Id;
769     cmd.Parameters[1].Value = checklistItem.Id;
770
771     //Execute sql statement
772     cmd.Prepare();
773     result = cmd.ExecuteNonQuery() == 1;
774
775     //Close database
776     DB.GetConnection().Close();
777     return result;
778 }
779 public static bool AddUserToUserStory(User user, UserStory userStory)
780 {
781
782     //Initialize variables
783     OleDbCommand cmd;
784     bool result;
785
786     //Open database, build sql statement and prepare
787     DB.GetConnection().Open();
788     cmd = DB.GetConnection().CreateCommand();
789     cmd.CommandText = "INSERT INTO TUserUserStory(IdUser, IdUserStory) ←
790                     VALUES(?,?);";
791     cmd.Parameters.Add("IdUser", OleDbType.Integer);
792     cmd.Parameters.Add("IdProject", OleDbType.Integer);
793     cmd.Parameters[0].Value = user.Id;
794     cmd.Parameters[1].Value = userStory.Id;
795
796     //Execute sql statement
797     cmd.Prepare();
798     result = cmd.ExecuteNonQuery() == 1;
799
800     //Close database
801     DB.GetConnection().Close();
802     return result;
803 }
804 public static bool AddUserToProject(User user, Project project)
805 {
806     //Initialize variables
807     OleDbCommand cmd;
808     bool result;
809
810     //Open database, build sql statement and prepare
811     DB.GetConnection().Open();
812     cmd = DB.GetConnection().CreateCommand();
813     cmd.CommandText = "INSERT INTO TUserProject(IdUser, IdProject) ←
814                     VALUES(?,?);";
815     cmd.Parameters.Add("IdUser", OleDbType.Integer);
816     cmd.Parameters.Add("IdProject", OleDbType.Integer);
817     cmd.Parameters[0].Value = user.Id;
818     cmd.Parameters[1].Value = project.Id;
819
820     //Execute sql statement
821     cmd.Prepare();
822     result = cmd.ExecuteNonQuery() == 1;
823
824     //Close database
825     DB.GetConnection().Close();
826     return result;
827 }
828 public static bool AddUserStoryToSprint(UserStory userStory, Sprint ←
829                                         sprint, int order)
830 {
831     //Initialize variables
832     OleDbCommand cmd;
833     bool result;
834
835     //Open database, build sql statement and prepare
836     DB.GetConnection().Open();
837     cmd = DB.GetConnection().CreateCommand();
838     cmd.CommandText = "INSERT INTO TUserStoriesSprint(IdUserStory, ←
839                      IdSprint, OrderUserStory) VALUES(?, ?, ?);";
840
841     cmd.Parameters.Add("IdUserStory", OleDbType.Integer);
842     cmd.Parameters.Add("IdSprint", OleDbType.Integer);
843     cmd.Parameters.Add("OrderUserStory", OleDbType.Integer);
844
845     cmd.Parameters[0].Value = userStory.Id;

```

```

842     cmd.Parameters[1].Value = sprint.Id;
843     cmd.Parameters[2].Value = order;
844
845     //Execute sql statement
846     cmd.Prepare();
847     result = cmd.ExecuteNonQuery() == 1;
848
849     //Close database
850     DB.GetConnection().Close();
851     return result;
852 }
853
854 #endregion
855 #region REMOVE
856 //Set the deletion flag to true
857 public static bool Delete(Activity activity)
858 {
859     //Initialize variables
860     OleDbCommand cmd;
861     bool result;
862
863     //Open database, build sql statement and prepare
864     DB.GetConnection().Open();
865     cmd = DB.GetConnection().CreateCommand();
866     cmd.CommandText = "UPDATE [TActivities] SET deletedFlag=? WHERE [IdActivity]=?";
867     cmd.Parameters.Add("deletedFlag", OleDbType.Boolean);
868     cmd.Parameters.Add("IdActivity", OleDbType.Integer);
869     cmd.Parameters[0].Value = true;
870     cmd.Parameters[1].Value = activity.Id;
871
872     //Execute sql statement
873     cmd.Prepare();
874     result = cmd.ExecuteNonQuery() == 1;
875
876     //Close database
877     DB.GetConnection().Close();
878     return result;
879 }
880 public static bool Delete(ChecklistItem checklistItem)
881 {
882     //Initialize variables
883     OleDbCommand cmd;
884     bool result;
885
886     //Open database, build sql statement and prepare
887     DB.GetConnection().Open();
888     cmd = DB.GetConnection().CreateCommand();
889     cmd.CommandText = "UPDATE [TChecklistItems] SET deletedFlag=? WHERE [IdChecklistItem]=?";
890     cmd.Parameters.Add("deletedFlag", OleDbType.Boolean);
891     cmd.Parameters.Add("IdChecklistItem", OleDbType.Integer);
892     cmd.Parameters[0].Value = true;
893     cmd.Parameters[1].Value = checklistItem.Id;
894
895     //Execute sql statement
896     cmd.Prepare();
897     result = cmd.ExecuteNonQuery() == 1;
898
899     //Close database
900     DB.GetConnection().Close();
901     return result;
902 }
903 public static bool Delete(Checklist checklist)
904 {
905     //Initialize variables
906     OleDbCommand cmd;
907     bool result;
908
909     //Open database, build sql statement and prepare
910     DB.GetConnection().Open();
911     cmd = DB.GetConnection().CreateCommand();
912     cmd.CommandText = "UPDATE [TChecklists] SET deletedFlag=? WHERE [IdChecklist]=?";
913     cmd.Parameters.Add("deletedFlag", OleDbType.Boolean);
914     cmd.Parameters.Add("IdChecklist", OleDbType.Integer);
915     cmd.Parameters[0].Value = true;
916     cmd.Parameters[1].Value = checklist.Id;
917 }
```

```

918     //Execute sql statement
919     cmd.Prepare();
920     result = cmd.ExecuteNonQuery() == 1;
921 
922     //Close database
923     DB.GetConnection().Close();
924     return result;
925 }
926 public static bool Delete(Comment comment)
927 {
928     //Initialize variables
929     OleDbCommand cmd;
930     bool result;
931 
932     //Open database, build sql statement and prepare
933     DB.GetConnection().Open();
934     cmd = DB.GetConnection().CreateCommand();
935     cmd.CommandText = "UPDATE [TComments] SET deletedFlag=? WHERE IdComment=?";
936     cmd.Parameters.Add("deletedFlag", OleDbType.Boolean);
937     cmd.Parameters.Add("IdComment", OleDbType.Integer);
938     cmd.Parameters[0].Value = true;
939     cmd.Parameters[1].Value = comment.Id;
940 
941     //Execute sql statement
942     cmd.Prepare();
943     result = cmd.ExecuteNonQuery() == 1;
944 
945     //Close database
946     DB.GetConnection().Close();
947     return result;
948 }
949 public static bool Delete(Classes.File File)
950 {
951     //Initialize variables
952     OleDbCommand cmd;
953     bool result;
954 
955     //Open database, build sql statement and prepare
956     DB.GetConnection().Open();
957     cmd = DB.GetConnection().CreateCommand();
958     cmd.CommandText = "UPDATE [TFiles] SET deletedFlag=? WHERE IdFile=?";
959     cmd.Parameters.Add("deletedFlag", OleDbType.Boolean);
960     cmd.Parameters.Add("IdFile", OleDbType.Integer);
961     cmd.Parameters[0].Value = true;
962     cmd.Parameters[1].Value = File.Id;
963 
964     //Execute sql statement
965     cmd.Prepare();
966     result = cmd.ExecuteNonQuery() == 1;
967 
968     //Close database
969     DB.GetConnection().Close();
970     return result;
971 }
972 public static bool Delete(Project project)
973 {
974     //Initialize variables
975     OleDbCommand cmd;
976     bool result;
977 
978     //Open database, build sql statement and prepare
979     DB.GetConnection().Open();
980     cmd = DB.GetConnection().CreateCommand();
981     cmd.CommandText = "UPDATE [TProjects] SET deletedFlag=? WHERE IdProject=?";
982     cmd.Parameters.Add("deletedFlag", OleDbType.Boolean);
983     cmd.Parameters.Add("IdProject", OleDbType.Integer);
984     cmd.Parameters[0].Value = true;
985     cmd.Parameters[1].Value = project.Id;
986 
987     //Execute sql statement
988     cmd.Prepare();
989     result = cmd.ExecuteNonQuery() == 1;
990 
991     //Close database
992     DB.GetConnection().Close();
993     return result;

```

```

994     }
995     public static bool Delete(Sprint sprint)
996     {
997         //Initialize variables
998         OleDbCommand cmd;
999         bool result;
1000
1001         //Open database, build sql statement and prepare
1002         DB.GetConnection().Open();
1003         cmd = DB.GetConnection().CreateCommand();
1004         cmd.CommandText = "UPDATE [TSpints] SET deletedFlag = ? WHERE IdSprint = ?;";
1005         cmd.Parameters.Add("deletedFlag", OleDbType.Boolean);
1006         cmd.Parameters.Add("IdSprint", OleDbType.Integer);
1007         cmd.Parameters[0].Value = true;
1008         cmd.Parameters[1].Value = sprint.Id;
1009
1010         //Execute sql statement
1011         cmd.Prepare();
1012         result = cmd.ExecuteNonQuery() == 1;
1013
1014         //Close database
1015         DB.GetConnection().Close();
1016         return result;
1017     }
1018     public static bool Delete(State state)
1019     {
1020         //Initialize variables
1021         OleDbCommand cmd;
1022         bool result;
1023
1024         //Open database, build sql statement and prepare
1025         DB.GetConnection().Open();
1026         cmd = DB.GetConnection().CreateCommand();
1027         cmd.CommandText = "UPDATE [TStates] SET deletedFlag = ? WHERE IdState = ?;";
1028         cmd.Parameters.Add("deletedFlag", OleDbType.Boolean);
1029         cmd.Parameters.Add("IdState", OleDbType.Integer);
1030         cmd.Parameters[0].Value = true;
1031         cmd.Parameters[1].Value = state.Id;
1032
1033         //Execute sql statement
1034         cmd.Prepare();
1035         result = cmd.ExecuteNonQuery() == 1;
1036
1037         //Close database
1038         DB.GetConnection().Close();
1039         return result;
1040     }
1041     public static bool Delete(User user)
1042     {
1043         //Initialize variables
1044         OleDbCommand cmd;
1045         bool result;
1046
1047         //Open database, build sql statement and prepare
1048         DB.GetConnection().Open();
1049         cmd = DB.GetConnection().CreateCommand();
1050         cmd.CommandText = "UPDATE [TUsers] SET deletedFlag = ? WHERE IdUser = ?;";
1051         cmd.Parameters.Add("deletedFlag", OleDbType.Boolean);
1052         cmd.Parameters.Add("IdUser", OleDbType.Integer);
1053         cmd.Parameters[0].Value = true;
1054         cmd.Parameters[1].Value = user.Id;
1055
1056         //Execute sql statement
1057         cmd.Prepare();
1058         result = cmd.ExecuteNonQuery() == 1;
1059
1060         //Close database
1061         DB.GetConnection().Close();
1062         return result;
1063     }
1064     public static bool Delete(UserStory userStory)
1065     {
1066         //Initialize variables
1067         OleDbCommand cmd;
1068         bool result;
1069

```

```

1070 //Open database, build sql statement and prepare
1071 DB.GetConnection().Open();
1072 cmd = DB.GetConnection().CreateCommand();
1073 cmd.CommandText = "UPDATE [TUserStories] SET deletedFlag=? WHERE [IdUserStory]=?";
1074 cmd.Parameters.Add("deletedFlag", OleDbType.Boolean);
1075 cmd.Parameters.Add("IdUserStory", OleDbType.Integer);
1076 cmd.Parameters[0].Value = true;
1077 cmd.Parameters[1].Value = userStory.Id;
1078
1079 //Execute sql statement
1080 cmd.Prepare();
1081 result = cmd.ExecuteNonQuery() == 1;
1082
1083 //Close database
1084 DB.GetConnection().Close();
1085 return result;
1086 }
1087 public static bool RemoveUserFromUserStory(User user, UserStory userStory)
1088 {
1089 //Initialize variables
1090 OleDbCommand cmd;
1091 bool result;
1092
1093 //Open database, build sql statement and prepare
1094 DB.GetConnection().Open();
1095 cmd = DB.GetConnection().CreateCommand();
1096 cmd.CommandText = "DELETE FROM [TUserUserStory] WHERE [IdUser]=? AND [IdUserStory]=?";
1097 cmd.Parameters.Add("IdUser", OleDbType.Integer);
1098 cmd.Parameters.Add("IdUserStory", OleDbType.Integer);
1099 cmd.Parameters[0].Value = user.Id;
1100 cmd.Parameters[1].Value = userStory.Id;
1101
1102 //Execute sql statement
1103 cmd.Prepare();
1104 result = cmd.ExecuteNonQuery() == 1;
1105
1106 //Close database
1107 DB.GetConnection().Close();
1108 return result;
1109 }
1110 public static bool RemoveUserFromProject(User user, Project project)
1111 {
1112 //Initialize variables
1113 OleDbCommand cmd;
1114 bool result;
1115
1116 //Open database, build sql statement and prepare
1117 DB.GetConnection().Open();
1118 cmd = DB.GetConnection().CreateCommand();
1119 cmd.CommandText = "DELETE FROM [TUserProject] WHERE [IdUser]=? AND [IdProject]=?";
1120 cmd.Parameters.Add("IdUser", OleDbType.Integer);
1121 cmd.Parameters.Add("IdProject", OleDbType.Integer);
1122 cmd.Parameters[0].Value = user.Id;
1123 cmd.Parameters[1].Value = project.Id;
1124
1125 //Execute sql statement
1126 cmd.Prepare();
1127 result = cmd.ExecuteNonQuery() == 1;
1128
1129 //Close database
1130 DB.GetConnection().Close();
1131 return result;
1132 }
1133 public static bool RemoveUserFromChecklistItem(User user, ChecklistItem checklistItem)
1134 {
1135 //Initialize variables
1136 OleDbCommand cmd;
1137 bool result;
1138
1139 //Open database, build sql statement and prepare
1140 DB.GetConnection().Open();
1141 cmd = DB.GetConnection().CreateCommand();
1142 cmd.CommandText = "DELETE FROM [TUserChecklistItem] WHERE [IdUser]=? AND [IdChecklistItem]=?";
1143 cmd.Parameters.Add("IdUser", OleDbType.Integer);

```

```

1144     cmd.Parameters.Add("IdChecklistItem", OleDbType.Integer);
1145     cmd.Parameters[0].Value = user.Id;
1146     cmd.Parameters[1].Value = checklistItem.Id;
1147
1148     //Execute sql statement
1149     cmd.Prepare();
1150     result = cmd.ExecuteNonQuery() == 1;
1151
1152     //Close database
1153     DB.GetConnection().Close();
1154     return result;
1155 }
1156 public static bool RemoveUserStoryFromSprint(UserStory userStory, ←
1157     Sprint sprint, int order)
1158 {
1159     //Initialize variables
1160     OleDbCommand cmd;
1161     bool result;
1162
1163     //Open database, build sql statement and prepare
1164     DB.GetConnection().Open();
1165     cmd = DB.GetConnection().CreateCommand();
1166     cmd.CommandText = "DELETE FROM TUserStoriesSprint WHERE IdUserStory←
1167         = ? AND IdSprint = ? AND OrderUserStory = ?;";
1168
1169     cmd.Parameters.Add("IdUserStory", OleDbType.Integer);
1170     cmd.Parameters.Add("IdSprint", OleDbType.Integer);
1171     cmd.Parameters.Add("OrderUserStory", OleDbType.Integer);
1172
1173     cmd.Parameters[0].Value = userStory.Id;
1174     cmd.Parameters[1].Value = sprint.Id;
1175     cmd.Parameters[2].Value = order;
1176
1177     //Execute sql statement
1178     cmd.Prepare();
1179     result = cmd.ExecuteNonQuery() == 1;
1180
1181     //Close database
1182     DB.GetConnection().Close();
1183     return result;
1184 }
1185 public static bool RemoveStateFromProject(Project project, int order)
1186 {
1187     //Initialize variables
1188     OleDbCommand cmd;
1189     bool result;
1190
1191     //Open database, build sql statement and prepare
1192     DB.GetConnection().Open();
1193     cmd = DB.GetConnection().CreateCommand();
1194     cmd.CommandText = "DELETE FROM TProjectStates WHERE IdProject = ?←
1195         AND orderState = ?;";
1196     cmd.Parameters.Add("IdProject", OleDbType.Integer);
1197     cmd.Parameters.Add("orderState", OleDbType.Integer);
1198     cmd.Parameters[0].Value = project.Id;
1199     cmd.Parameters[1].Value = order;
1200
1201     //Execute sql statement
1202     cmd.Prepare();
1203     result = cmd.ExecuteNonQuery() == 1;
1204
1205     //Close database
1206     DB.GetConnection().Close();
1207     return result;
1208 }
1209 public static bool Delete(MindMap mindmap)
1210 {
1211     //Initialize variables
1212     OleDbCommand cmd;
1213     bool result;
1214
1215     //Open database, build sql statement and prepare
1216     DB.GetConnection().Open();
1217     cmd = DB.GetConnection().CreateCommand();
1218     cmd.CommandText = "UPDATE TMindMaps SET deletedFlag = ? WHERE IdMindMap = ?;";
1219     cmd.Parameters.Add("deletedFlag", OleDbType.Boolean);
1220     cmd.Parameters.Add("IdMindMap", OleDbType.Integer);
1221     cmd.Parameters[0].Value = true;

```

```

1219         cmd.Parameters[1].Value = mindmap.Id;
1220
1221     //Execute sql statement
1222     cmd.Prepare();
1223     result = cmd.ExecuteNonQuery() == 1;
1224
1225     //Close database
1226     DB.GetConnection().Close();
1227     return result;
1228 }
1229 public static bool Delete(Node node)
1230 {
1231     if (node.PreviousId == null)
1232     {
1233         return false;
1234     }
1235     //Initialize variables
1236     OleDbCommand cmd;
1237     bool result;
1238
1239     //Open database, build sql statement and prepare
1240     DB.GetConnection().Open();
1241     cmd = DB.GetConnection().CreateCommand();
1242     cmd.CommandText = "UPDATE [TNodes] SET [deletedFlag] = ? WHERE [IdNode] = ?";
1243     cmd.Parameters.Add("deletedFlag", OleDbType.Boolean);
1244     cmd.Parameters.Add("IdNode", OleDbType.Integer);
1245     cmd.Parameters[0].Value = true;
1246     cmd.Parameters[1].Value = node.Id;
1247
1248     //Execute sql statement
1249     cmd.Prepare();
1250     result = cmd.ExecuteNonQuery() == 1;
1251
1252     //Close database
1253     DB.GetConnection().Close();
1254     return result;
1255 }
1256 #endregion
1257 #region GET
1258 //get objects with deletedflag on false
1259 public static List<int[]> GetUserStoriesSprint()
1260 {
1261     //Declare variables
1262     OleDbCommand cmd;
1263     OleDbDataReader rdr;
1264     object[] values;
1265     List<int[]> valuesPair = new List<int[]>();
1266
1267     //Open Database
1268     DB.GetConnection().Open();
1269     cmd = DB.GetConnection().CreateCommand();
1270
1271     //Execute SQL Command
1272     cmd.CommandText = "SELECT * FROM [TUserStoriesSprint];";
1273
1274     //Read and assign states with project
1275     rdr = cmd.ExecuteReader();
1276     values = new object[3];
1277     while (rdr.Read())
1278     {
1279         rdr.GetValues(values);
1280         // 0: IdUserStory, 1: IdSprint, 2: Order
1281         valuesPair.Add(new int[] { (int)values[0], (int)values[1], (int)values[2] });
1282     }
1283
1284     //Close database and reader
1285     rdr.Close();
1286     DB.GetConnection().Close();
1287
1288     return valuesPair;
1289 }
1290 public static List<int[]> GetProjectStates()
1291 {
1292     //Declare variables
1293     OleDbCommand cmd;
1294     OleDbDataReader rdr;
1295     object[] values;

```

```

1296     List<int []> valuesPair = new List<int []>();
1297
1298     //Open Database
1299     DB.GetConnection().Open();
1300     cmd = DB.GetConnection().CreateCommand();
1301
1302     //Execute SQL Command
1303     cmd.CommandText = "SELECT * FROM TProjectStates;";
1304
1305     //Read and assign states with project
1306     rdr = cmd.ExecuteReader();
1307     values = new object[3];
1308     while (rdr.Read())
1309     {
1310         rdr.GetValues(values);
1311         // 0: IdProject, 1: IdState, 2: order
1312         valuesPair.Add(new int[] { (int)values[0], (int)values[1], (int)values[2] });
1313     }
1314
1315     //Close database and reader
1316     rdr.Close();
1317     DB.GetConnection().Close();
1318
1319     return valuesPair;
1320 }
1321 public static List<int []> GetUserProject()
1322 {
1323     //Declare variables
1324     OleDbCommand cmd;
1325     OleDbDataReader rdr;
1326     object[] values;
1327     List<int []> valuesPair = new List<int []>();
1328
1329     //Open Database
1330     DB.GetConnection().Open();
1331     cmd = DB.GetConnection().CreateCommand();
1332
1333     //Execute SQL Command
1334     cmd.CommandText = "SELECT * FROM TUserProject;";
1335
1336     //Read and assign states with project
1337     rdr = cmd.ExecuteReader();
1338     values = new object[2];
1339     while (rdr.Read())
1340     {
1341         rdr.GetValues(values);
1342         // 0: IdUser, 1: IdProject
1343         valuesPair.Add(new int[] { (int)values[0], (int)values[1] });
1344     }
1345
1346     //Close database and reader
1347     rdr.Close();
1348     DB.GetConnection().Close();
1349
1350     return valuesPair;
1351 }
1352 public static List<int []> GetUserChecklistItem()
1353 {
1354     //Declare variables
1355     OleDbCommand cmd;
1356     OleDbDataReader rdr;
1357     object[] values;
1358     List<int []> valuesPair = new List<int []>();
1359
1360     //Open Database
1361     DB.GetConnection().Open();
1362     cmd = DB.GetConnection().CreateCommand();
1363
1364     //Execute SQL Command
1365     cmd.CommandText = "SELECT * FROM TUserChecklistItem;";
1366
1367     //Read and assign states with project
1368     rdr = cmd.ExecuteReader();
1369     values = new object[2];
1370     while (rdr.Read())
1371     {
1372         rdr.GetValues(values);
1373         // 0: IdUser, 1: IdChecklistItem

```

```

1374         valuesPair.Add(new int[] { (int)values[0], (int)values[1] });
1375     }
1376 
1377     //Close database and reader
1378     rdr.Close();
1379     DB.GetConnection().Close();
1380 
1381     return valuesPair;
1382 }
1383 public static List<int[]> GetUserUserStory()
1384 {
1385     //Declare variables
1386     OleDbCommand cmd;
1387     OleDbDataReader rdr;
1388     object[] values;
1389     List<int[]> valuesPair = new List<int[]>();
1390 
1391     //Open Database
1392     DB.GetConnection().Open();
1393     cmd = DB.GetConnection().CreateCommand();
1394 
1395     //Execute SQL Command
1396     cmd.CommandText = "SELECT * FROM TUserUserStory";
1397 
1398     //Read and assign states with project
1399     rdr = cmd.ExecuteReader();
1400     values = new object[2];
1401     while (rdr.Read())
1402     {
1403         rdr.GetValues(values);
1404         // 0: IdUser , 1: IdUserStory
1405         valuesPair.Add(new int[] { (int)values[0], (int)values[1] });
1406     }
1407 
1408     //Close database and reader
1409     rdr.Close();
1410     DB.GetConnection().Close();
1411 
1412     return valuesPair;
1413 }
1414 public static List<Project> GetProjects()
1415 {
1416     //Declare variables
1417     OleDbCommand cmd;
1418     OleDbDataReader rdr;
1419     object[] values;
1420     List<Project> projects = new List<Project>();
1421 
1422     //Open Database
1423     DB.GetConnection().Open();
1424     cmd = DB.GetConnection().CreateCommand();
1425 
1426     //Execute SQL Command
1427     cmd.CommandText = "SELECT * FROM TProjects WHERE deletedFlag = FALSE";
1428     cmd.Connection = DB.GetConnection();
1429 
1430     //Read and put projects in a list
1431     rdr = cmd.ExecuteReader();
1432     values = new object[4];
1433     while (rdr.Read())
1434     {
1435         rdr.GetValues(values);
1436         // 0: IdProject , 1: NameProject , 2: Description , 3: DateBegin
1437         Project p = new Project((int)values[0], (string)values[1], 
1438             (string)values[2], (DateTime)values[3]);
1439         projects.Add(p);
1440     }
1441 
1442     //Close database and reader
1443     rdr.Close();
1444     GetConnection().Close();
1445 
1446     return projects;
1447 }
1448 public static List<Sprint> GetSprints()
1449 {
1450     //Declare variables
1451     OleDbCommand cmd;

```

```

1451     OleDbDataReader rdr;
1452     object[] values;
1453     List<Sprint> sprints = new List<Sprint>();
1454
1455     //Open Database
1456     DB.GetConnection().Open();
1457     cmd = DB.GetConnection().CreateCommand();
1458
1459     //Execute SQL Command
1460     cmd.CommandText = "SELECT * FROM TSprints WHERE deletedFlag = FALSE ;";
1461     cmd.Connection = DB.GetConnection();
1462
1463     //Read and put entries in a list of objects
1464     rdr = cmd.ExecuteReader();
1465     values = new object[4];
1466     while (rdr.Read())
1467     {
1468         rdr.GetValues(values);
1469         // 0:idSprint, 1:DateBegin, 2:DateEnd
1470         Sprint s = new Sprint((int)values[0], (DateTime)values[1], ←
1471             (DateTime)values[2], (int)values[3]);
1472         sprints.Add(s);
1473     }
1474
1475     //Close database and reader
1476     rdr.Close();
1477     GetConnection().Close();
1478
1479     return sprints;
1480 }
1481 public static List<UserStory> GetUserStories()
1482 {
1483     //Declare variables
1484     OleDbCommand cmd;
1485     OleDbDataReader rdr;
1486     object[] values;
1487     List<UserStory> userStories = new List<UserStory>();
1488
1489     //Open Database
1490     DB.GetConnection().Open();
1491     cmd = DB.GetConnection().CreateCommand();
1492
1493     //Execute SQL Command
1494     cmd.CommandText = "SELECT * FROM TUserStories WHERE deletedFlag = ←
1495         FALSE ;";
1496     cmd.Connection = DB.GetConnection();
1497
1498     //Read and put entries in a list of objects
1499     rdr = cmd.ExecuteReader();
1500     values = new object[10];
1501     while (rdr.Read())
1502     {
1503         rdr.GetValues(values);
1504         // 0:idUserStory, 1:DescriptionUserStory, 2:DateLimite, ←
1505             3:ComplexityEstimation, 4:CompletedComplexity, 5:Blocked, ←
1506             6:ProjectId, 7:StateId, 8:TypeId, 9:PriorityId
1507         UserStory u = new UserStory((int)values[0], (string)values[1], ←
1508             values[2] as DateTime?, (int)values[3], (int)values[4], ←
1509             (bool)values[5], (int)values[6], (int)values[7], ←
1510             (int)values[8], (int)values[9]);
1511         userStories.Add(u);
1512     }
1513
1514     //Close database and reader
1515     rdr.Close();
1516     GetConnection().Close();
1517
1518     return userStories;
1519 }
1520 public static List<User> GetUsers()
1521 {
1522     //Declare variables
1523     OleDbCommand cmd;
1524     OleDbDataReader rdr;
1525     object[] values;
1526     List<User> users = new List<User>();
1527
1528     //Open Database
1529     DB.GetConnection().Open();

```

```

1523     cmd = DB.GetConnection().CreateCommand();
1524
1525     //Execute SQL Command
1526     cmd.CommandText = "SELECT * FROM TUsers WHERE deletedFlag = FALSE ;";
1527     cmd.Connection = DB.GetConnection();
1528
1529     //Read and put entries in a list of objects
1530     rdr = cmd.ExecuteReader();
1531     values = new object[2];
1532     while (rdr.Read())
1533     {
1534         rdr.GetValues(values);
1535         // 0:idUser ,1:NameUser
1536         User u = new User((int)values[0], (string)values[1]);
1537         users.Add(u);
1538     }
1539
1540     //Close database and reader
1541     rdr.Close();
1542     GetConnection().Close();
1543
1544     return users;
1545 }
1546 public static List<Classes.Type> GetTypes()
1547 {
1548     //Declare variables
1549     OleDbCommand cmd;
1550     OleDbDataReader rdr;
1551     object[] values;
1552     List<Classes.Type> types = new List<Classes.Type>();
1553
1554     //Open Database
1555     DB.GetConnection().Open();
1556     cmd = DB.GetConnection().CreateCommand();
1557
1558     //Execute SQL Command
1559     cmd.CommandText = "SELECT * FROM TTTypes WHERE deletedFlag = FALSE ;";
1560     cmd.Connection = DB.GetConnection();
1561
1562     //Read and put entries in a list of objects
1563     rdr = cmd.ExecuteReader();
1564     values = new object[2];
1565     while (rdr.Read())
1566     {
1567         rdr.GetValues(values);
1568         // 0:idType ,1:NameType
1569         Classes.Type t = new Classes.Type((int)values[0], ←
1570             (string)values[1]);
1571         types.Add(t);
1572     }
1573
1574     //Close database and reader
1575     rdr.Close();
1576     GetConnection().Close();
1577
1578     return types;
1579 }
1580 public static List<Priority> GetPriorities()
1581 {
1582     //Declare variables
1583     OleDbCommand cmd;
1584     OleDbDataReader rdr;
1585     object[] values;
1586     List<Priority> priorities = new List<Priority>();
1587
1588     //Open Database
1589     DB.GetConnection().Open();
1590     cmd = DB.GetConnection().CreateCommand();
1591
1592     //Execute SQL Command
1593     cmd.CommandText = "SELECT * FROM TPriorities WHERE deletedFlag = ←
1594         FALSE ;";
1595     cmd.Connection = DB.GetConnection();
1596
1597     //Read and put entries in a list of objects
1598     rdr = cmd.ExecuteReader();
1599     values = new object[2];
1600     while (rdr.Read())
1601     {

```

```

1600         rdr.GetValues(values);
1601         // 0:idPriority ,1:NamePriority
1602         Priority u = new Priority((int)values[0], (string)values[1]);
1603         priorities.Add(u);
1604     }
1605
1606     //Close database and reader
1607     rdr.Close();
1608     GetConnection().Close();
1609
1610     return priorities;
1611 }
1612 public static List<Classes.File> GetFiles()
1613 {
1614     //Declare variables
1615     OleDbCommand cmd;
1616     OleDbDataReader rdr;
1617     object[] values;
1618     List<Classes.File> files = new List<Classes.File>();
1619
1620     //Open Database
1621     DB.GetConnection().Open();
1622     cmd = DB.GetConnection().CreateCommand();
1623
1624     //Execute SQL Command
1625     cmd.CommandText = "SELECT * FROM TFiles WHERE deletedFlag = FALSE ;";
1626     cmd.Connection = DB.GetConnection();
1627
1628     //Read and put entries in a list of objects
1629     rdr = cmd.ExecuteReader();
1630     values = new object[4];
1631     while (rdr.Read())
1632     {
1633         rdr.GetValues(values);
1634         // 0:idFile ,1:nameFile ,2:DescFile ,3:idUserStory
1635         Classes.File f = new Classes.File((int)values[0], ←
1636             (string)values[1], (string)values[2], (int)values[3]);
1637         files.Add(f);
1638     }
1639
1640     //Close database and reader
1641     rdr.Close();
1642     GetConnection().Close();
1643
1644     return files;
1645 }
1646 public static List<Activity> GetActivities()
1647 {
1648     //Declare variables
1649     OleDbCommand cmd;
1650     OleDbDataReader rdr;
1651     object[] values;
1652     List<Activity> activities = new List<Activity>();
1653
1654     //Open Database
1655     DB.GetConnection().Open();
1656     cmd = DB.GetConnection().CreateCommand();
1657
1658     //Execute SQL Command
1659     cmd.CommandText = "SELECT * FROM TAactivities WHERE deletedFlag = ←
1660         FALSE ;";
1661     cmd.Connection = DB.GetConnection();
1662
1663     //Read and put entries in a list of objects
1664     rdr = cmd.ExecuteReader();
1665     values = new object[4];
1666     while (rdr.Read())
1667     {
1668         rdr.GetValues(values);
1669         // 0:idFileType ,1:description ,2:DateTimeActivity ,3:IdUserStory
1670         Activity a = new Activity((int)values[0], (string)values[1], ←
1671             (DateTime)values[2], (int)values[3]);
1672         activities.Add(a);
1673     }
1674
1675     //Close database and reader
1676     rdr.Close();
1677     GetConnection().Close();
1678 }
```

```

1676         return activities;
1677     }
1678     public static List<Checklist> GetChecklists()
1679     {
1680         //Declare variables
1681         OleDbCommand cmd;
1682         OleDbDataReader rdr;
1683         object[] values;
1684         List<Checklist> checklists = new List<Checklist>();
1685
1686         //Open Database
1687         DB.GetConnection().Open();
1688         cmd = DB.GetConnection().CreateCommand();
1689
1690         //Execute SQL Command
1691         cmd.CommandText = "SELECT * FROM TChecklists WHERE deletedFlag = FALSE";
1692         cmd.Connection = DB.GetConnection();
1693
1694         //Read and put entries in a list of objects
1695         rdr = cmd.ExecuteReader();
1696         values = new object[3];
1697         while (rdr.Read())
1698         {
1699             rdr.GetValues(values);
1700             // 0:idChecklist,1:description ,2:idUserStory
1701             Checklist c = new Checklist((int)values[0], (string)values[1], (int)values[2]);
1702             checklists.Add(c);
1703         }
1704
1705         //Close database and reader
1706         rdr.Close();
1707         GetConnection().Close();
1708
1709         return checklists;
1710     }
1711     public static List<ChecklistItem> GetChecklistItems()
1712     {
1713         //Declare variables
1714         OleDbCommand cmd;
1715         OleDbDataReader rdr;
1716         object[] values;
1717         List<ChecklistItem> checklistItems = new List<ChecklistItem>();
1718
1719         //Open Database
1720         DB.GetConnection().Open();
1721         cmd = DB.GetConnection().CreateCommand();
1722
1723         //Execute SQL Command
1724         cmd.CommandText = "SELECT * FROM TChecklistItems WHERE deletedFlag = FALSE";
1725         cmd.Connection = DB.GetConnection();
1726
1727         //Read and put entries in a list of objects
1728         rdr = cmd.ExecuteReader();
1729         values = new object[4];
1730         while (rdr.Read())
1731         {
1732             rdr.GetValues(values);
1733             // 0:idChecklistItem,1:nameItem,2:done ,3:idChecklist
1734             ChecklistItem c = new ChecklistItem((int)values[0], (string)values[1], (bool)values[2], (int)values[3]);
1735             checklistItems.Add(c);
1736         }
1737
1738         //Close database and reader
1739         rdr.Close();
1740         GetConnection().Close();
1741
1742         return checklistItems;
1743     }
1744     public static List<Comment> GetComments()
1745     {
1746         //Declare variables
1747         OleDbCommand cmd;
1748         OleDbDataReader rdr;
1749         object[] values;
1750         List<Comment> comments = new List<Comment>();

```

```

1751
1752     //Open Database
1753     DB.GetConnection().Open();
1754     cmd = DB.GetConnection().CreateCommand();
1755
1756     //Execute SQL Command
1757     cmd.CommandText = "SELECT * FROM TComments WHERE deletedFlag = FALSE;";
1758     cmd.Connection = DB.GetConnection();
1759
1760     //Read and put entries in a list of objects
1761     rdr = cmd.ExecuteReader();
1762     values = new object[5];
1763     while (rdr.Read())
1764     {
1765         rdr.GetValues(values);
1766         // 0:idComment,1:desc,2:dateTime,3:idUserStory,4:idUser
1767         Comment c = new Comment((int)values[0], (string)values[1], (DateTime)values[2], (int)values[3], (int)values[4]);
1768         comments.Add(c);
1769     }
1770
1771     //Close database and reader
1772     rdr.Close();
1773     GetConnection().Close();
1774
1775     return comments;
1776 }
1777 public static List<State> GetStates()
1778 {
1779     //Declare variables
1780     OleDbCommand cmd;
1781     OleDbDataReader rdr;
1782     object[] values;
1783     List<State> states = new List<State>();
1784
1785     //Open Database
1786     DB.GetConnection().Open();
1787     cmd = DB.GetConnection().CreateCommand();
1788
1789     //Execute SQL Command
1790     cmd.CommandText = "SELECT * FROM TStates WHERE deletedFlag = FALSE;";
1791     cmd.Connection = DB.GetConnection();
1792
1793     //Read and put entries in a list of objects
1794     rdr = cmd.ExecuteReader();
1795     values = new object[2];
1796     while (rdr.Read())
1797     {
1798         rdr.GetValues(values);
1799         // 0:IdState,1:NameState
1800         states.Add(new State((int)values[0], (string)values[1]));
1801     }
1802
1803     //Close database and reader
1804     rdr.Close();
1805     GetConnection().Close();
1806
1807     return states;
1808 }
1809 public static List<MindMap> GetMindMaps()
1810 {
1811     //Declare variables
1812     OleDbCommand cmd;
1813     OleDbDataReader rdr;
1814     object[] values;
1815     List<MindMap> mindmaps = new List<MindMap>();
1816
1817     //Open Database
1818     DB.GetConnection().Open();
1819     cmd = DB.GetConnection().CreateCommand();
1820
1821     //Execute SQL Command
1822     cmd.CommandText = "SELECT * FROM TMindMaps WHERE deletedFlag = FALSE;";
1823     cmd.Connection = DB.GetConnection();
1824
1825     //Read and put entries in a list of objects

```

```

1827     rdr = cmd.ExecuteReader();
1828     values = new object[4];
1829     while (rdr.Read())
1830     {
1831         rdr.GetValues(values);
1832         // 0:idMindMap ,1:NameMindMap ,2:idProject ,3:deletedFlag
1833         MindMap m = new MindMap((int)values[0], (string)values[1], ←
1834             (int)values[2]);
1835         mindmaps.Add(m);
1836     }
1837 
1838     //Close database and reader
1839     rdr.Close();
1840     GetConnection().Close();
1841 
1842     return mindmaps;
1843 }
1844 public static List<Node> GetNodes()
{
    //Declare variables
    OleDbCommand cmd;
    OleDbDataReader rdr;
    object[] values;
    List<Node> nodes = new List<Node>();

    //Open Database
    DB.GetConnection().Open();
    cmd = DB.GetConnection().CreateCommand();

    //Execute SQL Command
    cmd.CommandText = "SELECT * FROM TNodes WHERE deletedFlag = FALSE";
    cmd.Connection = DB.GetConnection();

    //Read and put entries in a list of objects
    rdr = cmd.ExecuteReader();
    values = new object[5];
    while (rdr.Read())
    {
        rdr.GetValues(values);

        if (values[2] == DBNull.Value)
        {
            values[2] = null;
        }
        // 0:idNode ,1:NameNode ,2:PreviousIdNode ,3:idMindmap ,4:deletedFlag
        Node n = new Node((int)values[0], (string)values[1], ←
            (int?)values[2], (int)values[3]);
        nodes.Add(n);
    }

    //Close database and reader
    rdr.Close();
    GetConnection().Close();

    return nodes;
}
#endregion
}
}

```

Listing 84 – ../../Scrum'o'wall/Scrum'o'wall/Controls/DB.cs

5 Tests

5.1 ControllerTests.cs

```
1  /*
2   * Author    :   Gaël Serge Mariot
3   * Project   :   Scrum'o'wall
4   * File      :   ControllerTests.cs
5   * Desc.     :   This file test the Controller class
6   */
7  using Microsoft.VisualStudio.TestTools.UnitTesting;
8  using Scrum_o_wall.Classes;
9  using System;
10 using System.Collections.Generic;
11 using System.Linq;
12
13 namespace Scrum_o_wall.Tests
14 {
15     [TestClass()]
16     public class ControllerTests
17     {
18         public static Controller ctrl;
19         [TestInitialize]
20         public void TestInitialize()
21         {
22             DB.DbFileName = @"C:\Users\redwo\OneDrive\Scrum-o-Wall\TestDB.accdb";
23
24             ctrl = new Controller();
25         }
26
27
28         [TestMethod]
29         public void ControllerTest()
30         {
31             Assert.IsNotNull(ctrl);
32         }
33
34         [TestMethod()]
35         public void CRDUserStoriesSprintTest()
36         {
37             ctrl.CreateProject("a name for a project", "a desc for a project", ←
38                 DateTime.Now);
39             Project project = ctrl.Projects.Last();
40
41             ctrl.CreateSprint(DateTime.Now, DateTime.Now, project);
42             Sprint sprint = project.Sprints[0];
43
44             //if not exists, the database is incorrect
45             Priority prio = ctrl.Priorities[0];
46             Classes.Type type = ctrl.Types[0];
47             ctrl.CreateUserStory("a description", null, 2, prio, type, project);
48             UserStory userStory = project.AllUserStories[0];
49
50             Assert.IsTrue(ctrl.AddUserStoryToSprint(userStory, sprint));
51             Assert.IsTrue(sprint.OrderedUserStories.ContainsValue(userStory));
52             Assert.IsTrue(ctrl.RemoveUserStoryFromSprint(userStory, sprint));
53
54             ctrl.Delete(userStory);
55             ctrl.Delete(sprint);
56             ctrl.Delete(project);
57         }
58         [TestMethod()]
59         public void CRDProjectStatesTest()
60         {
61             ctrl.CreateProject("aProjectName", "A Description for my project", ←
62                 DateTime.Now);
63             Project project = ctrl.Projects.Last();
64
65             ctrl.CreateState("a new state");
66             State state = ctrl.States.Last();
67
68             Assert.IsTrue(ctrl.AddStateToProject(state, project));
69             Assert.IsTrue(project.States.ContainsValue(state));
70             Assert.IsTrue(ctrl.RemoveStateFromProject(state, project));
71
72             ctrl.Delete(project);
73             ctrl.Delete(state);
74         }
75     }
76 }
```

```

73 }
74
75 [TestMethod]
76 public void CRDUserIUsersAssignedTest()
77 {
78     ctrl.CreateProject("aProjectName", "A\u201dDescription\u201dfor\u201dmy\u201dproject", ←
79         DateTime.Now);
80     Project project = ctrl.Projects.Last();
81
82     ctrl.CreateUser("a\u201duser\u201dname", project);
83     User user = ctrl.Users.Last();
84
85     Assert.IsTrue(ctrl.AddUserToIUsersAssigned(user, project));
86     Assert.IsTrue(project.GetUsers().Contains(user));
87     Assert.IsTrue(ctrl.RemoveUserFromIUsersAssigned(user, project));
88
89
90     ctrl.CreateUserStory("aDescription", null, 2, ctrl.Priorities[0], ←
91         ctrl.Types[0], project);
92     UserStory userStory = project.AllUserStories[0];
93
94     Assert.IsTrue(ctrl.AddUserToIUsersAssigned(user, userStory));
95     Assert.IsTrue(userStory.GetUsers().Contains(user));
96     Assert.IsTrue(ctrl.RemoveUserFromIUsersAssigned(user, userStory));
97
98     Checklist checklist = ctrl.CreateCheckList("a\u201dbioutifoul\u201dname", ←
99         userStory);
100    ctrl.CreateCheckListItem("a\u201dbetter\u201dname", checklist);
101    ChecklistItem checklistItem = checklist.ChecklistItems[0];
102
103    Assert.IsTrue(ctrl.AddUserToIUsersAssigned(user, checklistItem));
104    Assert.IsTrue(checklistItem.GetUsers().Contains(user));
105    Assert.IsTrue(ctrl.RemoveUserFromIUsersAssigned(user, checklistItem));
106
107    ctrl.Delete(user);
108    ctrl.Delete(checklistItem);
109    ctrl.Delete(checklist);
110    ctrl.Delete(userStory);
111    ctrl.Delete(project);
112 }
113 [TestMethod]
114 public void CRDActivity()
115 {
116     ctrl.CreateProject("aProjectName", "A\u201dDescription\u201dfor\u201dmy\u201dproject", ←
117         DateTime.Now);
118     Project project = ctrl.Projects.Last();
119     ctrl.CreateUserStory("aDescription", null, 2, ctrl.Priorities[0], ←
120         ctrl.Types[0], project);
121     UserStory userStory = project.AllUserStories[0];
122     string aDesc = "a\u201desc\u201dfor\u201dmy\u201dactivity";
123
124     Assert.IsTrue(ctrl.CreateActivity(aDesc, userStory));
125     Activity activity = userStory.Activities.Last();
126     Assert.AreEqual(aDesc, activity.Description);
127     Assert.IsTrue(ctrl.Delete(activity));
128
129     ctrl.Delete(userStory);
130     ctrl.Delete(project);
131 }
132
133 [TestMethod]
134 public void CRUDChecklist()
135 {
136     //Test Create
137     ctrl.CreateProject("aProjectName", "A\u201dDescription\u201dfor\u201dmy\u201dproject", ←
138         DateTime.Now);
139     Project project = ctrl.Projects.Last();
140     ctrl.CreateUserStory("aDescription", null, 2, ctrl.Priorities[0], ←
141         ctrl.Types[0], project);
142     UserStory userStory = project.AllUserStories[0];
143     string aName = "a\u201dfirst\u201dname";
144
145     Checklist checklist = ctrl.CreateCheckList(aName, userStory);
146
147     Assert.AreEqual(aName, checklist.Name);
148     Assert.AreEqual(userStory.Id, checklist.UserStoryId);
149
150     //Test Update
151     string afterName = "checklist\u201dsecond\u201dname";

```

```

145     Assert.IsTrue(ctrl.UpdateCheckList(afterName, new ←
146                     List<ChecklistItem>(), checklist));
147
148     Assert.AreEqual(afterName, checklist.Name);
149
150     //Test Remove
151     Assert.IsTrue(ctrl.Delete(checklist));
152
153     ctrl.Delete(userStory);
154     ctrl.Delete(project);
155 }
156 [TestMethod]
157 public void CRUDChecklistItem()
158 {
159     //Test Create
160     string aName = "checklistItem\u00b7first\u00b7name";
161     ctrl.CreateProject("aProjectName", "A\u00b7Description\u00b7for\u00b7my\u00b7project", ←
162                     DateTime.Now);
163     Project project = ctrl.Projects.Last();
164     ctrl.CreateUserStory("aDescription", null, 2, ctrl.Priorities[0], ←
165                         ctrl.Types[0], project);
166     UserStory userStory = project.AllUserStories[0];
167     ctrl.CreateCheckList("aCheck", userStory);
168     Checklist checklist = userStory.Checklists[0];
169
170     Assert.IsTrue(ctrl.CreateCheckListItem(aName, checklist));
171     ChecklistItem checklistItem = checklist.ChecklistItems[0];
172
173     Assert.AreEqual(aName, checklistItem.NameItem);
174     Assert.IsFalse(checklistItem.Done);
175     Assert.AreEqual(checklist, checklistItem.Checklist);
176
177     //Test Update
178     string afterName = "checklistItem\u00b7second\u00b7name";
179
180     Assert.IsTrue(ctrl.UpdateCheckListItem(afterName, true, ←
181                     checklistItem));
182
183     Assert.AreEqual(afterName, checklistItem.NameItem);
184     Assert.IsTrue(checklistItem.Done);
185
186     //Test Remove
187     Assert.IsTrue(ctrl.Delete(checklistItem));
188
189     ctrl.Delete(checklist);
190     ctrl.Delete(userStory);
191     ctrl.Delete(project);
192 }
193 [TestMethod]
194 public void CRDComment()
195 {
196     //Test Create
197     string aName = "comment\u00b7first\u00b7name";
198     ctrl.CreateProject("aProjectName", "A\u00b7Description\u00b7for\u00b7my\u00b7project", ←
199                     DateTime.Now);
200     Project project = ctrl.Projects.Last();
201     ctrl.CreateUserStory("aDescription", null, 2, ctrl.Priorities[0], ←
202                         ctrl.Types[0], project);
203     UserStory userStory = project.AllUserStories[0];
204
205     ctrl.CreateUser("a\u00b7user\u00b7name", userStory);
206     User user = ctrl.Users.Last();
207
208     Assert.IsTrue(ctrl.CreateComment(aName, user, userStory));
209     Comment comment = userStory.Comments[0];
210
211     Assert.AreEqual(aName, comment.Description);
212     Assert.AreEqual(userStory.Id, comment.UserStoryId);
213     Assert.AreEqual(user.Id, comment.UserId);
214
215     Assert.IsTrue(ctrl.Delete(comment));
216
217     ctrl.Delete(user);
218     ctrl.Delete(userStory);
219     ctrl.Delete(project);
220 }
221 [TestMethod]
222 public void CRUDFile()

```

```

218     {
219         //Declare needed variables
220         string afterDesc = "file\u00a9second\u00a9name";
221         string aName = "file\u00a9first\u00a9name";
222         string aDesc = "this\u00a9is\u00a9a\u00a9description";
223         ctrl.CreateProject("aProjectName", "A\u00a9Description\u00a9for\u00a9my\u00a9project", ←
224             DateTime.Now);
225         Project project = ctrl.Projects.Last();
226         ctrl.CreateUserStory("aDescription", null, 2, ctrl.Priorities[0], ←
227             ctrl.Types[0], project);
228         UserStory userStory = project.AllUserStories[0];
229
230         //Test Create
231         Assert.IsTrue(ctrl.CreateFile(aName, aDesc, userStory));
232         File file = userStory.Files[0];
233
234         Assert.AreEqual(aName, file.Name);
235         Assert.AreEqual(aDesc, file.Description);
236         Assert.AreEqual(userStory.Id, file.UserStoryId);
237
238         //Test Update
239         Assert.IsTrue(ctrl.UpdateFile(afterDesc, file));
240         Assert.AreEqual(afterDesc, file.Description);
241
242         //Test Remove
243         Assert.IsTrue(ctrl.Delete(file));
244
245         ctrl.Delete(project);
246         ctrl.Delete(userStory);
247     }
248     [TestMethod]
249     public void CRUDProject()
250     {
251         string firstName = "the\u00a9project\u00a9first\u00a9Name";
252         string firstDesc = "the\u00a9project\u00a9first\u00a9description";
253         DateTime firstDate = DateTime.Now;
254         string secName = "the\u00a9project\u00a9sec\u00a9Name";
255         string secDesc = "the\u00a9project\u00a9sec\u00a9description";
256         DateTime secDate = DateTime.Now + new TimeSpan(7, 0, 0, 0);
257
258         //Test Create
259         Assert.IsTrue(ctrl.CreateProject(firstName, firstDesc, firstDate));
260         Project project = ctrl.Projects.Last();
261
262         Assert.AreEqual(firstName, project.Name);
263         Assert.AreEqual(firstDesc, project.Description);
264         Assert.AreEqual(firstDate.ToString(), project.Begin.ToString());
265
266         //Test Update
267         Assert.IsTrue(ctrl.UpdateProject(secName, secDesc, secDate, project));
268
269         Assert.AreEqual(secName, project.Name);
270         Assert.AreEqual(secDesc, project.Description);
271         Assert.AreEqual(secDate.ToString(), project.Begin.ToString());
272
273         //Test Remove
274         Assert.IsTrue(ctrl.Delete(project));
275     }
276     [TestMethod]
277     public void CRUDSprint()
278     {
279         DateTime firstBegin = DateTime.Now;
280         DateTime firstEnd = firstBegin + new TimeSpan(7, 0, 0, 0);
281         ctrl.CreateProject("aProjectName", "A\u00a9Description\u00a9for\u00a9my\u00a9project", ←
282             DateTime.Now);
283         Project project = ctrl.Projects.Last();
284         DateTime secBegin = firstEnd;
285         DateTime secEnd = secBegin + new TimeSpan(7, 0, 0, 0);
286
287         //Test Create
288         Assert.IsTrue(ctrl.CreateSprint(firstBegin, firstEnd, project));
289         Sprint sprint = project.Sprints[0];
290
291         Assert.AreEqual(firstBegin.ToString(), sprint.Begin.ToString());
292         Assert.AreEqual(firstEnd.ToString(), sprint.End.ToString());
293
294         //Test Update
295
296         Assert.IsTrue(ctrl.UpdateSprint(secBegin, secEnd, sprint));

```

```

294
295     Assert.AreEqual(secBegin.ToString(), sprint.Begin.ToString());
296     Assert.AreEqual(secEnd.ToString(), sprint.End.ToString());
297
298     //Test Remove
299     Assert.IsTrue(ctrl.Delete(sprint));
300 }
301 [TestMethod]
302 public void CRUDState()
303 {
304     //Test Create
305     string firstName = "first\u00a5state\u00a5name";
306
307     Assert.IsTrue(ctrl.CreateState(firstName));
308     State state = ctrl.States.Last();
309
310     Assert.AreEqual(firstName, state.Name);
311
312
313     //Test Remove
314     Assert.IsTrue(ctrl.Delete(state));
315 }
316 [TestMethod]
317 public void CRUDUser()
318 {
319     //Test Create
320     string firstName = "first\u00a5user\u00a5name";
321
322     Assert.IsTrue(ctrl.CreateUser(firstName, null));
323     User user = ctrl.Users.Last();
324
325     Assert.AreEqual(firstName, user.Name);
326
327     //Test Remove
328     Assert.IsTrue(ctrl.Delete(user));
329 }
330 [TestMethod]
331 public void CRUDUserStoryTest()
332 {
333     ctrl.CreateProject("aProjectName", "A\u00a5Description\u00a5for\u00a5my\u00a5project", ←
334         DateTime.Now);
335     Project project = ctrl.Projects.Last();
336
337     string firstDesc = "aDesc";
338     DateTime? firstDate = DateTime.Now;
339     int firstComplexity = 2;
340     Priority firstPrio = ctrl.Priorities[0];
341     Classes.Type firstType = ctrl.Types[0];
342
343     Assert.IsTrue(ctrl.CreateUserStory(firstDesc, firstDate, ←
344         firstComplexity, firstPrio, firstType, project));
345     UserStory userStory = project.AllUserStories[0];
346
347     Assert.IsNotNull(userStory, "Exception\u00a5in\u00a5userStory\u00a5creation");
348     Assert.AreEqual(firstDesc, userStory.Description);
349     Assert.AreEqual(firstDate, userStory.DateLimit);
350     Assert.AreEqual(firstComplexity, userStory.ComplexityEstimation);
351     Assert.AreEqual(firstPrio, userStory.Priority);
352     Assert.AreEqual(firstType, userStory.Type);
353     Assert.AreEqual(project, userStory.Project);
354     Assert.AreEqual(false, userStory.Blocked);
355     Assert.AreEqual(0, userStory.CompletedComplexity);
356
357
358     string secDesc = "aNewDesc";
359     DateTime? secDate = null;
360     int secComplexity = 3;
361     int secCompleted = 1;
362     bool secBlock = true;
363     Priority secPrio = ctrl.Priorities[1];
364     Classes.Type secType = ctrl.Types[1];
365     while (ctrl.States.Count < 2)
366     {
367         ctrl.CreateState("An\u00a5additional\u00a5state\u00a5name");
368     }
369     State secState = ctrl.States.Last();

```

```

370     Assert.IsTrue(ctrl.UpdateUserStory(secDesc, secDate, secComplexity, ←
371                     secCompleted, secBlock, secPrio, secType, secState, userStory));
372 
373     Assert.AreEqual(secDesc, userStory.Description);
374     Assert.AreEqual(secDate, userStory.DateLimit);
375     Assert.AreEqual(secComplexity, userStory.ComplexityEstimation);
376     Assert.AreEqual(secPrio, userStory.Priority);
377     Assert.AreEqual(secType, userStory.Type);
378     Assert.AreEqual(secState, userStory.State);
379     Assert.AreEqual(secBlock, userStory.Blocked);
380     Assert.AreEqual(secCompleted, userStory.CompletedComplexity);
381 
382     Assert.IsTrue(ctrl.Delete(userStory));
383 
384     ctrl.Delete(project);
385 }
386 
387 [TestMethod]
388 public void CRUDMindMap()
389 {
390 
391     string firstName = "firstMindMap\u00b7name";
392     string secondName = "secondMindMap\u00b7name";
393     ctrl.CreateProject("a\u00b7project\u00b7name", "a\u00b7description", DateTime.Now);
394     Project project = ctrl.Projects.Last();
395 
396     //Test Create
397 
398     Assert.IsTrue(ctrl.CreateMindMap(firstName, project));
399     MindMap mindMap = project.MindMaps.Last();
400 
401     Assert.AreEqual(firstName, mindMap.Name);
402 
403     //Test Update
404     Assert.IsTrue(ctrl.UpdateMindMap(secondName, mindMap));
405     Assert.AreEqual(secondName, mindMap.Name);
406 
407     //Test Remove
408     Assert.IsTrue(ctrl.Delete(mindMap));
409 
410     ctrl.Delete(project);
411 }
412 
413 [TestMethod]
414 public void CRUDNode()
415 {
416 
417     ctrl.CreateProject("a\u00b7project\u00b7name", "a\u00b7description", DateTime.Now);
418     Project project = ctrl.Projects.Last();
419     ctrl.CreateMindMap("a\u00b7mind\u00b7map\u00b7name", project);
420     MindMap mindMap = project.MindMaps.Last();
421 
422     //Test Create Without Previous
423     string firstName = "first\u00b7node\u00b7name";
424 
425     Assert.IsTrue(ctrl.CreateNode(firstName, null, mindMap));
426     Node node = mindMap.Root;
427 
428     Assert.AreEqual(firstName, node.Name);
429     Assert.AreEqual(mindMap, node.MindMap);
430     //Test Create With Previous
431     string firstName2 = "first\u00b7node\u00b7name2";
432 
433     Assert.IsTrue(ctrl.CreateNode(firstName2, node, mindMap));
434     Node node2 = node.Childrens.Last();
435 
436     Assert.AreEqual(firstName2, node2.Name);
437     Assert.AreEqual(node, node2.Previous);
438     Assert.AreEqual(node2, node.Childrens.Last());
439 
440     //Test Update
441     string secondName = "second\u00b7node\u00b7name";
442 
443     Assert.IsFalse(ctrl.UpdateNode(secondName, node2, node));
444     Assert.IsTrue(ctrl.UpdateNode(secondName, null, node));
445     Assert.AreEqual(secondName, node.Name);
446 
447     //Test Remove
448     Assert.IsFalse(ctrl.Delete(node));
449     Assert.IsTrue(ctrl.Delete(node2));

```

```

448             ctrl.Delete(mindMap);
449             ctrl.Delete(project);
450         }
451     }
452 }
453 }
```

Listing 85 – ../../Scrum'o'wall/Scrum'o'wallTests/ControllerTests.cs

5.2 DBTests.cs

```

1  /*
2   * Author    :    Gael Serge Mariot
3   * Project   :    Scrum'o'wall
4   * File      :    DBTests.cs
5   * Desc.     :    This file test the DB class
6   */
7  using Microsoft.VisualStudio.TestTools.UnitTesting;
8  using Scrum_o_wall.Classes;
9  using System;
10 using System.Collections.Generic;
11 using System.Linq;
12
13 namespace Scrum_o_wall.Tests
14 {
15     [TestClass()]
16     public class DBTests
17     {
18         [TestInitialize]
19         public void TestInitialize()
20         {
21             DB.DbFileName = @"C:\Users\redwo\OneDrive\Scrum-o-Wall\TestDB.accdb";
22         }
23
24         [TestMethod()]
25         public void CRDUserStoriesSprintTest()
26         {
27             Project project = DB.CreateProject("aname", "adesc", DateTime.Now);
28
29             //Create UserStory
30             UserStory userStory;
31             List<Priority> priorities = DB.GetPriorities();
32             List<State> states = DB.GetStates();
33             List<Classes.Type> types = DB.GetTypes();
34             if (states.Count == 0)
35             {
36                 DB.CreateState("AStateName");
37                 states = DB.GetStates();
38             }
39             userStory = DB.CreateUserStory("aDescription", null, 2, ←
40                                         priorities[0], types[0], states[0], project);
41
42             //Create Sprint
43             Sprint sprint;
44             sprint = DB.CreateSprint(DateTime.Now, DateTime.Now, project);
45             int order = 0;
46
47             //Link ,test ,unlink
48             Assert.IsTrue(DB.AddUserStoryToSprint(userStory, sprint, order));
49             Assert.IsNotNull(DB.GetUserStoriesSprint());
50             Assert.IsTrue(DB.RemoveUserStoryFromSprint(userStory, sprint, order));
51
52             DB.Delete(project);
53             DB.Delete(userStory);
54             DB.Delete(sprint);
55         }
56         [TestMethod()]
57         public void CRDProjectStatesTest()
58         {
59             Project project = DB.CreateProject("aProjectName", "A\u00a0Description\u2014←
60                                         for\u00a0my\u00a0project", DateTime.Now);
61             State state = DB.CreateState("a\u00a0new\u00a0state");
62             int order = 0;
63
64             Assert.IsTrue(DB.AddStateToProject(state, project, order));
65             Assert.IsNotNull(DB.GetProjectStates());
66         }
67     }
68 }
```



```

137
138     //Create User
139     User user = DB.CreateUser("a\u00bduusername");
140
141     Assert.IsTrue(DB.AddUserToUserStory(user, userStory));
142     Assert.IsNotNull(DB.GetUserUserStory());
143     Assert.IsTrue(DB.RemoveUserFromUserStory(user, userStory));
144
145     DB.Delete(userStory);
146     DB.Delete(user);
147 }
148 [TestMethod()]
149 public void GetTypesTest()
150 {
151     Assert.IsNotNull(DB.GetTypes());
152 }
153 [TestMethod()]
154 public void GetPrioritiesTest()
155 {
156     Assert.IsNotNull(DB.GetPriorities());
157 }
158 [TestMethod]
159 public void CRDActivity()
160 {
161     string aDesc = "activity\u00bdt";
162     DateTime eventTime = DateTime.Now;
163     List<UserStory> userStories = DB.GetUserStories();
164     UserStory userStory;
165     if (userStories.Count == 0)
166     {
167         List<Priority> priorities = DB.GetPriorities();
168         List<State> states = DB.GetStates();
169         List<Classes.Type> types = DB.GetTypes();
170         List<Project> projects = DB.GetProjects();
171         if (states.Count == 0)
172         {
173             DB.CreateState("AStateName");
174             states = DB.GetStates();
175         }
176         if (projects.Count == 0)
177         {
178             DB.CreateProject("aProjectName", "A\u00bduDescription\u00bdufor\u00bdumy\u00bdu\u2192"
179                         project", DateTime.Now);
180             projects = DB.GetProjects();
181         }
182
183         userStory = DB.CreateUserStory("aDescription", null, 2, ←
184                                     priorities[0], types[0], states[0], projects[0]);
185     }
186     else
187     {
188         userStory = userStories.Last();
189     }
190
191     Activity a = DB.CreateActivity(aDesc, eventTime, userStory);
192
193     Assert.AreEqual(aDesc, a.Description);
194     Assert.AreEqual(eventTime, a.DateTime);
195     Assert.AreEqual(userStory.Id, a.UserStoryId);
196
197     Assert.IsNotNull(DB.GetActivities());
198
199     Assert.IsTrue(DB.Delete(a));
200 }
201 [TestMethod]
202 public void CRUDChecklist()
203 {
204     //Test Create
205     string aName = "checlist\u00bdfirst\u00bdufirstname";
206     List<UserStory> userStories = DB.GetUserStories();
207     UserStory userStory;
208     if (userStories.Count == 0)
209     {
210         List<Priority> priorities = DB.GetPriorities();
211         List<State> states = DB.GetStates();
212         List<Classes.Type> types = DB.GetTypes();
213         List<Project> projects = DB.GetProjects();

```

```

214         if (states.Count == 0)
215     {
216         DB.CreateState("AStateName");
217         states = DB.GetStates();
218     }
219     if (projects.Count == 0)
220     {
221         DB.CreateProject("aProjectName", "A Description for my ←
222             project", DateTime.Now);
223         projects = DB.GetProjects();
224     }
225
226     userStory = DB.CreateUserStory("aDescription", null, 2, ←
227         priorities[0], types[0], states[0], projects[0]);
228 }
229 else
230 {
231     userStory = userStories.Last();
232 }
233
234 Checklist checklist = DB.CreateCheckList(aName, userStory);
235
236 Assert.AreEqual(aName, checklist.Name);
237 Assert.AreEqual(userStory.Id, checklist.UserStoryId);
238
239 //Test Update
240 string afterName = "checklist second name";
241
242 DB.UpdateCheckList(afterName, checklist);
243 checklist = DB.GetChecklists().First(c => c.Id == checklist.Id);
244
245 Assert.AreEqual(afterName, checklist.Name);
246
247 //Test Remove
248 Assert.IsTrue(DB.Delete(checklist));
249
250 }
251 [TestMethod]
252 public void CRUDChecklistItem()
253 {
254     //Test Create
255     string aName = "checklistItem first name";
256     List<Checklist> checklists = DB.GetChecklists();
257     Checklist checklist;
258     if (checklists.Count == 0)
259     {
260         List<UserStory> userStories = DB.GetUserStories();
261         UserStory userStory;
262         if (userStories.Count == 0)
263         {
264             List<Priority> priorities = DB.GetPriorities();
265             List<State> states = DB.GetStates();
266             List<Classes.Type> types = DB.GetTypes();
267             List<Project> projects = DB.GetProjects();
268             if (states.Count == 0)
269             {
270                 DB.CreateState("AStateName");
271                 states = DB.GetStates();
272             }
273             if (projects.Count == 0)
274             {
275                 DB.CreateProject("aProjectName", "A Description for my ←
276                     project", DateTime.Now);
277                 projects = DB.GetProjects();
278             }
279
280             userStory = DB.CreateUserStory("aDescription", null, 2, ←
281                 priorities[0], types[0], states[0], projects[0]);
282         }
283     else
284     {
285         userStory = userStories.Last();
286     }
287
288     checklist = DB.CreateCheckList("the name of a checklist", ←
289         userStory);

```

```

288         }
289     }
290     else
291     {
292         checklist = checklists[0];
293     }
294     ChecklistItem checklistItem = DB.CreateCheckListItem(aName, ↪
295             checklist);
296
297     Assert.AreEqual(aName, checklistItem.NameItem);
298     Assert.IsFalse(checklistItem.Done);
299     Assert.AreEqual(checklist.Id, checklistItem.ChecklistId);
300
301     //Test Update
302     string afterName = "checklistItem\u00b7second\u00b7name";
303
304     DB.UpdateCheckListItem(afterName, true, checklistItem);
305     checklistItem = DB.GetChecklistItems().First(c => c.Id == ↪
306             checklistItem.Id);
307
308     Assert.AreEqual(afterName, checklistItem.NameItem);
309     Assert.IsTrue(checklistItem.Done);
310
311     //Test Remove
312     Assert.IsTrue(DB.Delete(checklistItem));
313 }
314 [TestMethod]
315 public void CRDComment()
316 {
317     //Test Create
318     string aName = "comment\u00b7first\u00b7name";
319     List<UserStory> userStories = DB.GetUserStories();
320     UserStory userStory;
321     if (userStories.Count == 0)
322     {
323         List<Priority> priorities = DB.GetPriorities();
324         List<State> states = DB.GetStates();
325         List<Classes.Type> types = DB.GetTypes();
326         List<Project> projects = DB.GetProjects();
327         if (states.Count == 0)
328         {
329             DB.CreateState("AStateName");
330             states = DB.GetStates();
331         }
332         if (projects.Count == 0)
333         {
334             DB.CreateProject("aProjectName", "A\u00b7Description\u00b7for\u00b7my\u00b7"
335                         project", DateTime.Now);
336             projects = DB.GetProjects();
337         }
338
339         userStory = DB.CreateUserStory("aDescription", null, 2, ↪
340             priorities[0], types[0], states[0], projects[0]);
341     }
342     else
343     {
344         userStory = userStories.Last();
345     }
346
347     User user = DB.GetUsers().Last();
348
349     Comment comment = DB.CreateComment(aName, userStory, user);
350
351     Assert.AreEqual(aName, comment.Description);
352     Assert.AreEqual(userStory.Id, comment.UserStoryId);
353     Assert.AreEqual(user.Id, comment.UserId);
354
355     Assert.IsNotNull(DB.GetComments());
356
357     //Test Remove
358     Assert.IsTrue(DB.Delete(comment));
359 }
360 [TestMethod]
361 public void CRUDFile()
362 {
363     //Test Create
364     string aName = "file\u00b7first\u00b7name";
365     string aDesc = "this\u00b7is\u00b7a\u00b7description";

```



```

440     DateTime firstBegin = DateTime.Now;
441     DateTime firstEnd = firstBegin + new TimeSpan(7, 0, 0, 0);
442     Project project = DB.GetProjects().Last();
443
444     Sprint sprint = DB.CreateSprint(firstBegin, firstEnd, project);
445
446     Assert.AreEqual(firstBegin.ToString(), sprint.Begin.ToString());
447     Assert.AreEqual(firstEnd.ToString(), sprint.End.ToString());
448
449     //Test Update
450     DateTime secBegin = firstEnd;
451     DateTime secEnd = secBegin + new TimeSpan(7, 0, 0, 0);
452
453     DB.UpdateSprint(secBegin, secEnd, sprint);
454     sprint = DB.GetSprints().First(s => s.Id == sprint.Id);
455
456     Assert.AreEqual(secBegin.ToString(), sprint.Begin.ToString());
457     Assert.AreEqual(secEnd.ToString(), sprint.End.ToString());
458
459     //Test Remove
460     Assert.IsTrue(DB.Delete(sprint));
461 }
462 [TestMethod]
463 public void CRUDState()
464 {
465     //Test Create
466     string firstName = "first\u00a5state\u00a5name";
467
468     State state = DB.CreateState(firstName);
469
470     Assert.AreEqual(firstName, state.Name);
471
472
473     //Test Remove
474     Assert.IsTrue(DB.Delete(state));
475 }
476 [TestMethod]
477 public void CRUDUser()
478 {
479     //Test Create
480     string firstName = "first\u00a5user\u00a5name";
481
482     User user = DB.CreateUser(firstName);
483
484     Assert.AreEqual(firstName, user.Name);
485
486     //Test Remove
487     Assert.IsTrue(DB.Delete(user));
488 }
489 [TestMethod]
490 public void CRUDUserStoryTest()
491 {
492     List<Priority> priorities = DB.GetPriorities();
493     List<State> states = DB.GetStates();
494     List<Classes.Type> types = DB.GetTypes();
495     List<Project> projects = DB.GetProjects();
496     if (states.Count == 0)
497     {
498         DB.CreateState("A\u00a5State\u00a5Name");
499         states = DB.GetStates();
500     }
501     if (projects.Count == 0)
502     {
503         DB.CreateProject("a\u00a5Project\u00a5Name", "A\u00a5Description\u00a5for\u00a5my\u00a5project", DateTime.Now);
504         projects = DB.GetProjects();
505     }
506
507     string firstDesc = "a\u00a5Desc";
508     DateTime? firstDate = DateTime.Now;
509     int firstComplexity = 2;
510     Priority firstPrio = priorities[0];
511     Classes.Type firstType = types[0];
512     State firstState = states[0];
513     Project firstProj = projects[0];
514
515
516     UserStory userStory = DB.CreateUserStory(firstDesc, firstDate, firstComplexity, firstPrio, firstType, firstState, firstProj);

```

```

517     Assert.IsNotNull(userStory, "Exception in userStory creation");
518     Assert.AreEqual(firstDesc, userStory.Description);
519     Assert.AreEqual(firstDate, userStory.DateLimit);
520     Assert.AreEqual(firstComplexity, userStory.ComplexityEstimation);
521     Assert.AreEqual(firstPrio.Id, userStory.PriorityId);
522     Assert.AreEqual(firstType.Id, userStory.TypeId);
523     Assert.AreEqual(firstState.Id, userStory.StateId);
524     Assert.AreEqual(firstProj.Id, userStory.ProjectId);
525     Assert.AreEqual(false, userStory.Blocked);
526     Assert.AreEqual(0, userStory.CompletedComplexity);
527
528
529     string secDesc = "aNewDesc";
530     DateTime? secDate = null;
531     int secComplexity = 3;
532     int secCompleted = 1;
533     bool secBlock = true;
534     Priority secPrio = priorities[1];
535     Classes.Type secType = types[1];
536     while (states.Count < 2)
537     {
538         DB.CreateState("An additional state name");
539         states = DB.GetStates();
540     }
541     State secState = states[1];
542
543     DB.UpdateUserStory(secDesc, secDate, secComplexity, secCompleted, ←
544                         secBlock, secPrio, secState, secType, userStory);
545
546     userStory = DB.GetUserStories().First(u => u.Id == userStory.Id);
547
548     Assert.AreEqual(secDesc, userStory.Description);
549     Assert.AreEqual(secDate, userStory.DateLimit);
550     Assert.AreEqual(secComplexity, userStory.ComplexityEstimation);
551     Assert.AreEqual(secPrio.Id, userStory.PriorityId);
552     Assert.AreEqual(secType.Id, userStory.TypeId);
553     Assert.AreEqual(secState.Id, userStory.StateId);
554     Assert.AreEqual(secBlock, userStory.Blocked);
555     Assert.AreEqual(secCompleted, userStory.CompletedComplexity);
556
557     Assert.IsTrue(DB.Delete(userStory));
558 }
559
560 [TestMethod]
561 public void CRUDMindMap()
562 {
563     Project project;
564
565     List<Project> projects = DB.GetProjects();
566     if (projects.Count == 0)
567     {
568         project = DB.CreateProject("a Project", "A desc of a project", ←
569                               DateTime.Now);
570     }
571     else
572     {
573         project = projects[0];
574     }
575
576     string firstName = "a mindmap first name";
577     string secondName = "a mindmap second name";
578
579     //Verify creation
580     MindMap mindMap = DB.CreateMindmap(firstName, project);
581     Assert.IsNotNull(mindMap);
582     Assert.AreEqual(firstName, mindMap.Name);
583     Assert.AreEqual(project.Id, mindMap.ProjectId);
584
585     //Verify update
586     Assert.IsTrue(DB.UpdateMindMap(secondName, mindMap));
587
588     //Verify delete
589     Assert.IsTrue(DB.Delete(mindMap));
590
591     DB.Delete(project);
592 }
593 [TestMethod]
594 public void CRUDNode()

```

```

594     {
595         Project project;
596         MindMap mindmap;
597
598         List<Project> projects = DB.GetProjects();
599         if (projects.Count == 0)
600         {
601             project = DB.CreateProject("a\u2014Project", "A\u2014desc\u2014of\u2014a\u2014project", ←
602             DateTime.Now);
603         }
604         else
605         {
606             project = projects[0];
607         }
608         List<MindMap> mindMaps = DB.GetMindMaps().Where(m => m.ProjectId == ←
609             project.Id).ToList();
610         if (mindMaps.Count == 0)
611         {
612             mindmap = DB.CreateMindmap("a\u2014mindmap\u2014name", project);
613         }
614         else
615         {
616             mindmap = mindMaps[0];
617
618             string firstName = "a\u2014node\u2014first\u2014name";
619             string firstName2 = "a\u2014second\u2014node\u2014first\u2014name";
620             string secondName = "a\u2014node\u2014second\u2014name";
621             string secondName2 = "a\u2014second\u2014node\u2014second\u2014name";
622
623             //Verify creation with and without previous
624             Node node = DB.CreateNode(firstName, null, mindmap);
625
626             Assert.IsNotNull(node);
627             Assert.IsNull(node.PreviousId);
628             Assert.AreEqual(firstName, node.Name);
629             Assert.AreEqual(mindmap.Id, node.MindmapId);
630
631             Node node2 = DB.CreateNode(firstName2, node, mindmap);
632
633             Assert.IsNotNull(node2);
634             Assert.AreEqual(node.Id, node2.PreviousId);
635             Assert.AreEqual(firstName2, node2.Name);
636             Assert.AreEqual(mindmap.Id, node2.MindmapId);
637
638             //Verify updates with and without previous
639             Assert.IsFalse(DB.UpdateNode(secondName, node2, node), "cannot\u2014←
640                 change\u2014previous\u2014to\u2014root");
641             Assert.IsTrue(DB.UpdateNode(secondName, null, node));
642             Assert.IsFalse(DB.UpdateNode(secondName2, null, node2), "cannot\u2014←
643                 change\u2014node\u2014to\u2014root");
644             Assert.IsTrue(DB.UpdateNode(secondName2, node, node2));
645
646             //Verify delete
647             Assert.IsFalse(DB.Delete(node), "cannot\u2014delete\u2014root");
648             Assert.IsTrue(DB.Delete(node2));
649
650             DB.Delete(mindmap);
651             DB.Delete(project);
652     }
}

```

Listing 86 – ../../Scrum'o'wall/Scrum'o'wallTests/DBTests.cs