



**LIFE PRO MANAGER**

***Barraud Laurent***

***Laurent.BARRAUD@cpnv.ch***

***Rossy David***

***David.ROSSY@cpnv.ch***

***Terrapon Julien***

***Julien.TERRAPON@cpnv.ch***

**SI-CA2a**

***Le 6 Novembre 2019***

Table des matières

[1 Introduction 3](#_Toc529773647)

[1.1 Cadre, description et motivation 3](#_Toc529773648)

[1.2 Organisation 3](#_Toc529773649)

[1.3 Objectifs 3](#_Toc529773650)

[1.4 Planification initiale 3](#_Toc529773651)

[2 Analyse 4](#_Toc529773652)

[2.1 Cahier des charges détaillé 4](#_Toc529773653)

[2.2 Stratégie de test 16](#_Toc529773654)

[2.3 Planification 17](#_Toc529773655)

[3 Implémentation 19](#_Toc529773656)

[3.1 Dossier de conception 19](#_Toc529773657)

[3.2 Dossier de réalisation 20](#_Toc529773658)

[3.3 Description des tests effectués 20](#_Toc529773659)

[4 Conclusions 20](#_Toc529773660)

[5 Annexes 21](#_Toc529773661)

[5.1 Sources – Bibliographie 21](#_Toc529773662)

# Introduction

## Cadre, description et motivation

Le projet est réalisé dans le cadre du module Projet(binôme) au CPNV. Nous avons choisi de développer une application Windows de gestion de tâches réalisée en C#. Ce travail nous apporte de l’expérience en programmation ainsi que celle quant à l’exploitation et l’utilisation d’une base de données.

**Organisation générale du projet**

Barraud Laurent, Laurent.BARRAUD@cpnv.ch

Rossy David, David.ROSSY@cpnv.ch

Terrapon *Julien, Julien.TERRAPON@cpnv.ch*

**Responsable de projet**

Andolfatto Frederique, Frederique.ANDOLFATTO@cpnv.ch

Viret Loic, Loic.VIRET@cpnv.ch

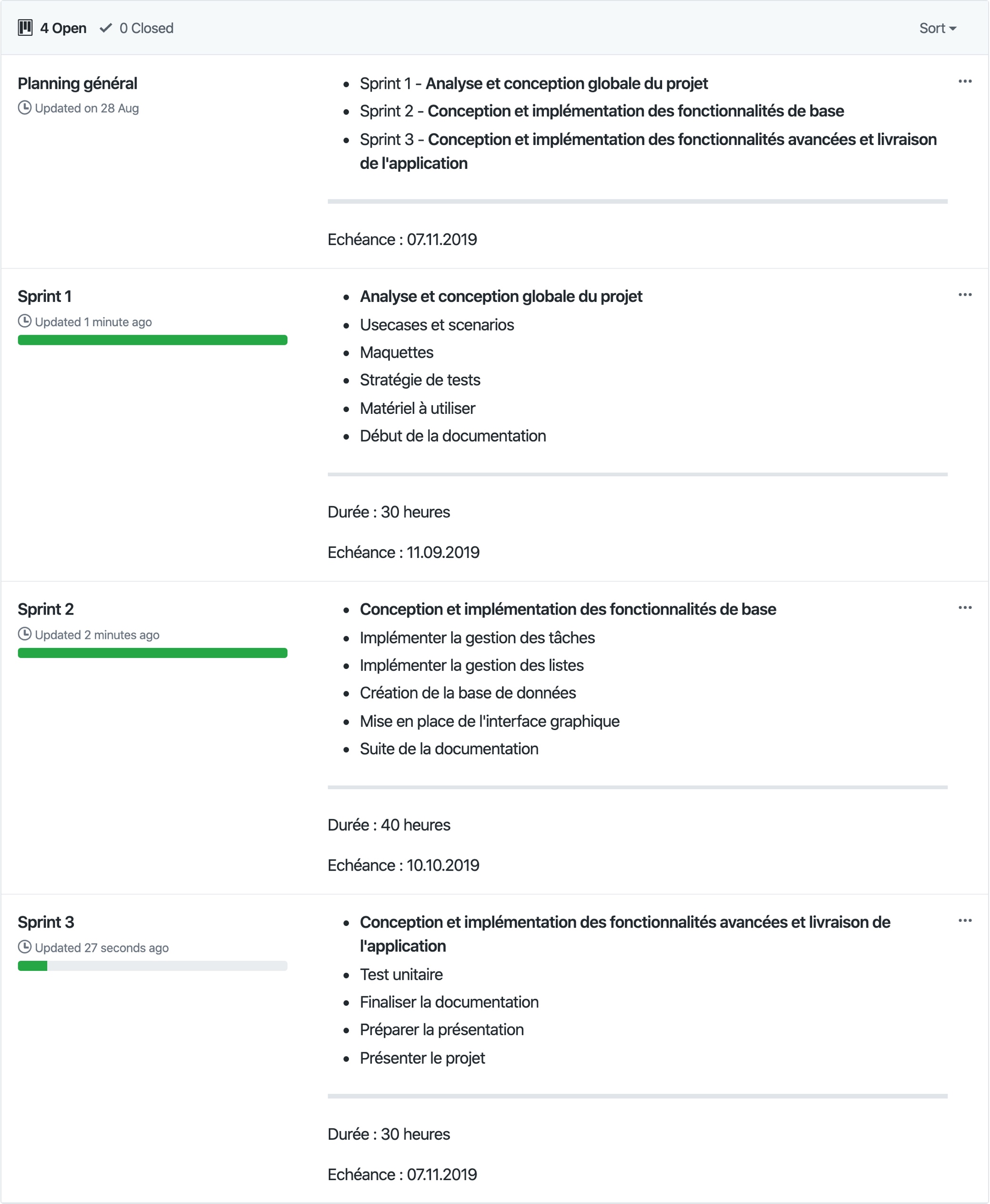
|  |  |  |
| --- | --- | --- |
|  | **F. Andolfatto**  **L. Barraud**  **D. Rossy**  **J. Terrapon** | **L. Viret** |
| Partie administration | X |  |
| Partie client |  | X |

## Objectifs

Au terme du premier trimestre, nous aurons développé une application de liste de tâches à raison de 4 heures de travail par semaine. En novembre, le codage des tâches (ajouter, modifier, supprimer, valider et dévalider) et des listes (ajouter, supprimer) sera terminé. D’ici là, l’application aura une interface user-friendly et ergonomique, sera fiable et réactive.

## Planification initiale

La planification se fera avec le site GitHub et se fera selon la méthode agile.



# Analyse

## Use cases et scénarios

1. Gérer des listes de tâches
   1. Créer une liste de tâches (thème)

|  |  |  |
| --- | --- | --- |
| **Action** | **Condition particulière** | **Réaction** |
| Appuyer sur l’onglet « Liste » |  | Accède à l’affichage des listes |
| Appuyer sur le bouton « Créer une nouvelle liste » |  | Ouvre une fenêtre de configuration des options de la liste |
| Configurer les options de la liste |  |  |
| Cliquer sur « OK » |  | Ferme la fenêtre de configuration de la liste et crée la liste |
| Cliquer sur « Annuler » |  | Ferme la fenêtre de configuration |

* 1. Attribuer une tâche à une liste

|  |  |  |
| --- | --- | --- |
| **Action** | **Condition particulière** | **Réaction** |
| Appuyer sur l’onglet « Liste » |  | Accède à l’affichage des listes |
| Appuyer sur le bouton « modifier la tâche » | La tâche est déjà créée et attribuée | Ouvre une fenêtre de configuration des options de la tâche |
| Créer la tâche (voir 3.1.1) | La tâche n’existe pas encore | Ouvre une fenêtre de configuration des options de la tâche |
| Cliquer sur la liste déroulante « Liste » |  | Déroule la liste des listes possible à l’attribution |
| Cliquer sur la liste voulue |  | Affiche la nouvelle liste à laquelle la tâche est attribuée |
| Cliquer sur « OK » |  | Ferme la fenêtre de configuration des options de la tâche |
| Cliquer sur « Annuler » |  | Ferme la fenêtre de configuration |

* 1. Supprimer une liste de tâches

|  |  |  |
| --- | --- | --- |
| **Action** | **Condition particulière** | **Réaction** |
| Appuyer sur le bouton « Liste » |  | Accède à l’affichage des listes |
| Appuyer sur le bouton « Supprimer la liste» de la liste correspondante |  | Ouvre une fenêtre de confirmation « Voulez-vous vraiment supprimer la liste ? Ceci entraînera également la suppression des tâches attribuées à celle-ci » |
| Cliquer sur « Oui » |  | Ferme la fenêtre de confirmation, supprime la liste ainsi que les tâches correspondantes |
| Cliquer sur « Non » |  | Ferme la fenêtre de confirmation |

1. Organiser des tâches dans des listes
   1. Créer une tâche

|  |  |  |
| --- | --- | --- |
| **Action** | **Condition particulière** | **Réaction** |
| Cliquer sur le bouton « Ajouter une tâche » |  | Une fenêtre s’ouvre avec les différentes données à entrer |
| Entrer les données |  |  |
| Cliquer sur le bouton « OK » | Tous les champs sont valides  Un ou plusieurs champs sont ne sont pas valides | Ajoute la tâche à la liste correspondante  Ouvre une boîte de dialogue avec un message d’erreur correspondant |
| Cliquer sur le bouton « Annuler » |  | La fenêtre se ferme |

* 1. Modifier une tâche

|  |  |  |
| --- | --- | --- |
| **Action** | **Condition particulière** | **Réaction** |
| Cliquer sur le bouton « Modifier » de la tâche correspondante |  | Ouvre une fenêtre de modification de la tâche |
| Modifier le-s champ-s | Les champs obligatoires sont remplis : titre, date de début et de fin, etc… | Les informations se modifient et retour sur la vue d’ensemble |
| Cliquer sur le bouton « OK » | Tous les champs sont valides  Un ou plusieurs champs sont ne sont pas valides | Modifie la tâche  Ouvre une boîte de dialogue avec un message d’erreur correspondant |
| Cliquer sur le bouton « Annuler » |  | La fenêtre se ferme |

* 1. Supprimer une tâche

|  |  |  |
| --- | --- | --- |
| **Action** | **Condition particulière** | **Réaction** |
| Cliquer sur le bouton « supprimer » de la tâche correspondante |  | Une fenêtre de confirmation s’ouvre « Etes-vous sûre de vouloir supprimer cette tâche ? » |
| Cliquer sur « Oui » |  | La tâche est supprimée de la liste |
| Cliquer sur « Non » |  | La fenêtre se ferme |

* 1. Changer le statut d’une tâche

|  |  |  |
| --- | --- | --- |
| **Action** | **Condition particulière** | **Réaction** |
| Cliquer sur la liste déroulante du statut de la tâche correspondante |  | La liste se déploie |
| Cliquer sur le statut désiré |  | La tâche change de statut |

* 1. Réduire un groupe

|  |  |  |
| --- | --- | --- |
| **Action** | **Condition particulière** | **Réaction** |
| L’utilisateur clique sur la flèche |  | Le groupe se réduit |
| L’utilisateur clique sur la flèche | Le groupe est réduit | Le groupe se déploie |

* 1. Prioriser les tâches

|  |  |  |
| --- | --- | --- |
| **Action** | **Condition particulière** | **Réaction** |
| Cliquer sur le bouton « Modifier » d’une tâche |  | Une fenêtre de modification s’ouvre |
| Cliquer sur la liste déroulante « Priorité » |  | La liste déroulante affiche les options |
| Sélectionner une priorité et cliquer sur « OK » |  | La priorité de la tâche est changée et la fenêtre se ferme |
| Cliquer sur « Annuler » |  | La fenêtre se ferme |

* 1. Afficher les tâches du jour

|  |  |  |
| --- | --- | --- |
| **Action** | **Condition particulière** | **Réaction** |
| Cliquer sur le bouton« Aujourd’hui » |  | L’affichage principal affiche les tâches du jour en détail |

* 1. Afficher la vue d’ensemble

|  |  |  |
| --- | --- | --- |
| **Action** | **Condition particulière** | **Réaction** |
| Cliquer sur « Vue d’ensemble » dans le menu |  | L’affichage principal affiche un condensé des tâches du jour ainsi que les tâches des 7 prochains jours |

* 1. Afficher les tâches d’un jour précis

|  |  |  |
| --- | --- | --- |
| **Action** | **Condition particulière** | **Réaction** |
| Cliquer sur un jour du calendrier |  | L’affichage principal affiche les tâches de ce jour en détails |

* 1. Être averti lorsqu’une tâche est en retard

|  |  |  |
| --- | --- | --- |
| **Action** | **Condition particulière** | **Réaction** |
|  | Une tâche arrive à échéance sans être terminée | Une notification apparaît à côté de la tâche ainsi qu’un message sur l’écran principal prévenant qu’une tâche est arrivée à échéance |
| Clique sur la notification |  | Une fenêtre contenant des détails sur la tâche en question apparaît |
| L’utilisateur clique sur le bouton « OK» |  | La fenêtre se ferme |

## Maquettes

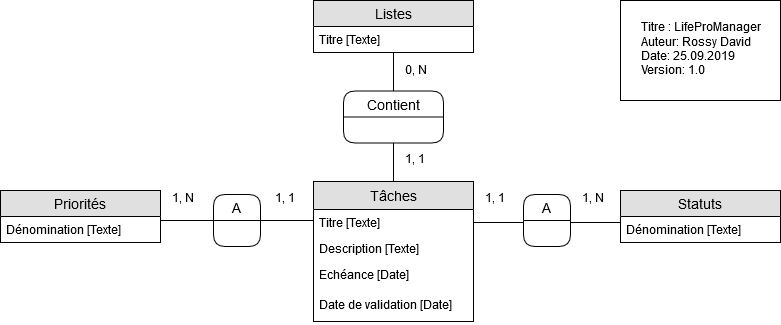
Les maquettes sont accessibles dans le répertoire de documentation

<https://github.com/DRossyCPNV/LifeProManager/tree/Documentation>

Ces dernières sont au format PDF cliquable afin d’améliorer la lisibilité.

Le panneau de gestion des tâches sur la droite s’est vu remplacé par des icônes à côté de chaque tâche afin de minimiser le nombre de cliques à effectuer.

## Modèle conceptuel de Données



## Stratégie de test

Nous utiliserons l’environnement de développement *Visual Studio 2017* sur *Windows 10*, installé sur chacun de nos ordinateurs respectifs de l’école, pour coder l’application *Life Pro Manager*. Ces postes nous serviront également à effectuer les tests fonctionnels ainsi que de robustesse.

Nous préparerons :

* Une base de données fictive contenant un grand nombre de tâches fictives (environ 100) ainsi qu’un grand nombre de listes (environ 50).
* Un script contenant les données fictives à insérer dans la base de données
* Un script de création de listes de tâches fictives

Nous déploierons ensuite une machine virtuelle *VMWare* fonctionnant sous *Windows 10*, également sur nos ordinateurs respectifs de l’école. Celle-ci servira à nous assurer que n’importe quel utilisateur disposant d’une machine sous *Windows 10*, sans nos programmes de développemen, pourra utiliser l’application.

**Point particulier**

Nous nous sommes heurtés à un problème de références manquantes lors de l’implémentation du test unitaire automatisé de la fonction d’ajout d’une liste/thème. Il s’agit d’un fichier « .dll » non trouvé du package SQLite (permettant d’intégrer une base de données locale à l’application).

Nous voulions vérifier la présence du thème à la liste déroulante de sélection des thèmes après l’ajout de ce dernier dans la base de données. Cependant nous redoutions que cela n’altère les données de la base, comportement que nous ne souhaitions pas. Nous avons donc fait une copie de test de la base. Le chemin de la base étant codé en dure, nous avons recopier une partie du code pour effectuer le test en référençant la base de test. Cette tentative s’est avérée sans succès.

Comme nous manquions de temps, nous nous sommes assurés du bon fonctionnement de la méthode par des tests manuels.

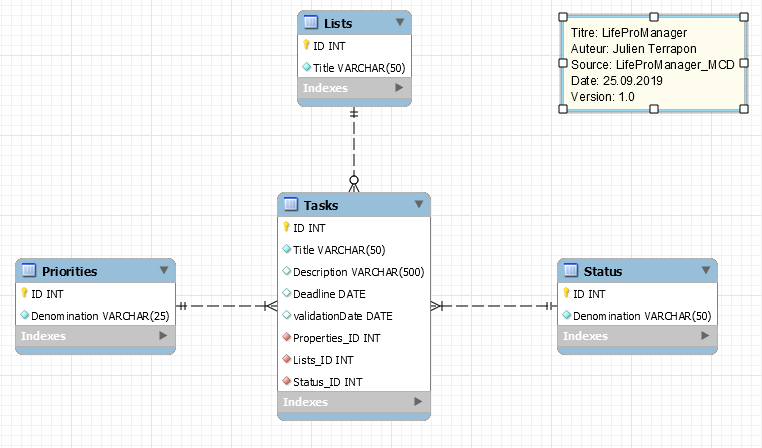
# Implémentation

## Choix techniques

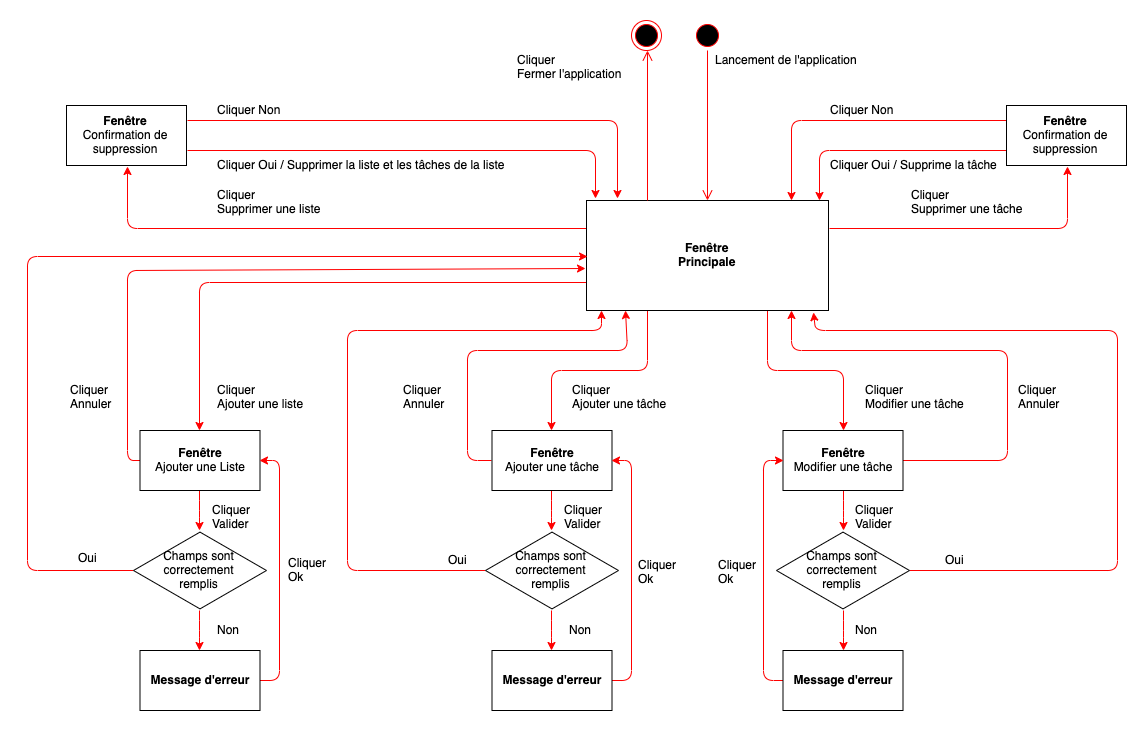
Ressources utilisées :

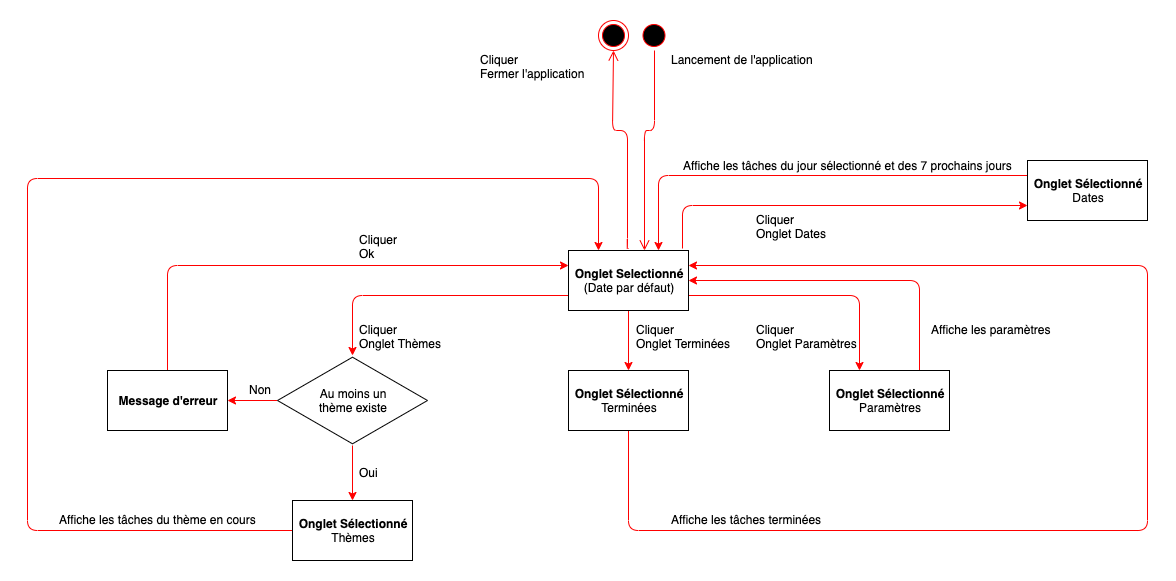
* **Windows 10 :** L’application est développée sous et pour cet OS.
* **Visual Studio Entreprise 2017 :** IDE idéal pour développer en C# et faire des interfaces graphiques avec des Windows Forms.
* **SQLite :** Base de données légère et portable idéale pour stocker les données d’une application locale n’ayant pas besoin d’un accès à distance
* **DB Browser for SQLite :** Outil visuel utilisé pour gérer et consulter la base de données SQLite en cours de développement.
* **Inno Setup Compiler :** Programme utilisé pour créer le fichier d’installation de l’application afin de faciliter le processus pour l’utilisateur.
* **Balsamiq :** Création des maquettes de l’application.
* **MySQL Workbench :** Logiciel utilisé pour concevoir le modèle logique de données.

## Modèle Logique de Données



## Dossier de conception





## Livraisons

Le livrable sera constitué de :

* Code source (Répertoire du projet Visual Studio)
* Un exécutable de l’application (Installeur pour utilisateur)
* Une documentation du code (Doxygen)
* Un guide d’installation
* Une documentation de projet

La date butoir est fixée au vendredi 8 novembre 2019, 23h59. Ce délais a été approuvé par l’équipe du projet ainsi que par le client.

# Tests

## Description des tests effectués

|  |  |
| --- | --- |
| Scénario | 08.11.2019  Développeur  Visual Studio 2017 / Windows 10  Version 1.0 |
| Créer une liste | **OK** |
| Supprimer une liste | **OK** |
| Ajouter une tâche sans avoir créé de liste | **OK** |
| Ajouter une tâche dans une liste | **OK** |
| Ne pas remplir les champs obligatoires dans la création d’une tâche | **OK** |
| Ne pas remplir les champs obligatoires lors de la modification d’une tâche | **OK** |
| Utiliser des caractères spéciaux lors de la saisie au clavier | **OK** |
| Modifier le titre d’une tâche | **OK** |
| Modifier la description d’une tâche | **OK** |
| Modifier l’échéance d’une tâche | **KO** |
| Modifier la priorité d’une tâche | **OK** |
| Modifier la liste d’une tâche | **OK** |
| Valider une tâche | **OK** |
| Dévalider une tâche | **OK** |
| Supprimer une tâche non-validée | **OK** |
| Supprimer une tâche validée | **OK** |

## Erreurs restantes

1. Crash de l’application lors de la saisie d’une date inexistante à l’ajout d’une tâche (ex : 31 novembre)

Utilisation ultérieure impossible pour l’utilisateur qui ne pourra pas relancer l’application (sans avoir corriger l’erreur dans la base de données), rendant le produit inutilisable sans réinstallation.

*Solution :*

*Nous prévoyions de résoudre ce problème en implémentant une fonction de vérification / validation de date avant de l’exploiter dans le programme.*

1. Lenteur possible à l’affichage lors du chargement d’un grand nombre de tâches

S’il ne s’agit pas d’une erreur ou d’un bug à proprement parler, on peut tout de même remarquer un léger temps de chargement lorsque l’on atteint la centaine de tâches. Il nous paraît important de le reporter ici, même si cette latence (de l’ordre de la seconde) dépendra en partie de la configuration de la machine de l’utilisateur.

*Solution :*

*Nous avons pensé à deux pistes pour résoudre ce désagrément :*

1. *Remanier le code en évitant de recharger les tâches sur tous les onglets à chaque ajout, validation, dévalidation, modification ou suppression de tâche.*
2. *Vérifier si le problème ne viendrait pas de l’omission d’un BEGIN TRANSACTION / COMMIT et corriger cela le cas échéant.*

# Conclusion

**Performance**

Nous sommes assez fiers du résultat de notre projet. Life Pro Manager est une application que l’on pourrait améliorer, si nous avions plus de temps, mais qui, à ce stade, a atteint tous les objectifs fixés. En effet, l’application est, selon nous, intuitive, rapide, légère et fait tout ce que l’on attend d’une todo liste.

**Planning**

Le planning est sans doute l’élément qui a été le moins respecté durant le projet. Ceci étant dû à une mauvaise estimation du temps de certaines tâches. Nous n’avons, par exemple, pas tout à fait respecté les principes de la méthode agile, en repoussant l’échéance du premier sprint review de deux semaines au lieu de réattribuer les tâches non-terminées au sprint suivant.

Bien que désagréables, ces mauvaises estimations nous auront appris à mieux évaluer la quantité de travail que requiert une tâche perçue comme rapide et facile de prime abord. Cela nous a également appris à avoir une vue plus globale du projet et de faire preuve de recul et d’accepter de re-prioriser ou abandonner certaines fonctionnalités.

**Communication**

Si l’équipe a travaillé dans la bonne entente et avec une répartition des tâches équitables, un bémol s’est tout de même fait ressentir au niveau de la validation des taches. Il nous est arrivé plusieurs fois de peiner à avancer en même temps dans l’implémentation de fonctionnalités interdépendantes. Nous nous sommes aussi quelques fois rendu compte que nous avions mal architecturé notre code, de l’ajuster en conséquence et d’impacter négativement une ou deux heures de travail de nos collègues.

Il nous est aussi arrivé d’avoir quelques problèmes de communication car si nous nous attribuions les tâches au début du jour, certains travaux ont été effectué à double par manque de concertation. Cela aurait pu être facilement évité si nous avions eu une plus grande rigueur en attribuant et consultant les tâches via l’outil de Github.

**Apprentissage**

Ce projet nous aura permis de mettre en pratique des concepts liés à la programmation orientée-objet. Certaines notions utiles ont été découvertes en cours alors que d’autres ont été le fruit de notre recherche personnelle pour le besoin de l’application. Nous avons beaucoup apprécier rechercher des solutions par nous-mêmes pour les besoins spécifiques d’une fonctionnalité. Le revers de la médaille est que ces solutions étaient parfois plus complexes que nécessaire. De ce fait notre code reste largement améliorable. Nous pensons particulièrement à la notion d’override d’une méthode, ToString() par exemple, qui nous aurait simplifier l’écriture de quelques fonctionnalités.

Un autre apprentissage que nous avons apprécié est celui d’intégrer une base de données à un programme. Nous avons ainsi pu exercer nos connaissances en langage SQL. Nous avons constaté comment pouvait être utilisé le langage dans un cas concret, en intégrant des variables, etc. et ce différemment qu’au travers d’un script SQL.

**Le projet plus loin**

Comme nous avions prévu beaucoup plus de fonctionnalités que ce que le temps nous permettait de réaliser, il est aisé d’imaginer une potentielle évolution. Nous aurions beaucoup aimé intégrer une notion d’utilisateur qui peut gérer des tâches, les partager avec d’autres, consulter les leurs, etc. Nous pourrions faire migrer notre base de données locale à l’application en une base de données commune stockée sur un serveur distant (peut-être MySQL). Life Pro Manager se muerait alors d’une simple application de bureau en une sorte d’application de gestion de tâches interactive/réseau social.

# Annexes

## Sources – Bibliographie

**Création d’un wizard d’installation**

<https://www.commentcamarche.net/faq/33511-creer-un-setup-d-installation-inno-setup-compiler>

**SQL pour SQLite**

<https://www.sqlite.org/lang_datefunc.html>

**Tableau composé d’objets différents**

<https://stackoverflow.com/questions/1932946/c-sharp-creating-array-where-the-array-value-has-multiple-objects-and-each-one>