

# TPI : Advanced To Do List



Théo Ghaemmaghami – FIN2

ETML - Lausanne

Du 13 mai 2024 au 3 juin 2024

Chef de projet : Monsieur Patrick Chenaux



## Table des matières

Remerciements .....	4
Introduction .....	5
1 Planification initiale .....	6
2 Analyse .....	17
2.1 Planification détaillée .....	17
2.2 Méthodologie .....	17
2.2.1 Méthode de gestion de projet .....	17
2.2.2 Sauvegarde .....	19
2.3 Conception .....	19
2.3.1 Modélisation de la base de données .....	19
2.3.2 UwAmp .....	21
2.3.3 Maquettes .....	21
2.3.4 Architecture MVC .....	25
2.4 Stratégie de test .....	26
3 Réalisation .....	27
3.1 Compte utilisateur .....	27
3.1.1 Hachage .....	27
3.1.2 Création de compte .....	28
3.1.3 Login (Connexion à un compte existant) .....	29
3.2 Gestion des tâches .....	30
3.2.1 Objet Tâche .....	30
3.2.2 Ajout d'une tâche .....	30
3.2.3 Affichage des tâches .....	30
4 Tests .....	31
5 Conclusion .....	32
6 Annexes .....	33
6.1 Table des illustrations .....	33

## Remerciements

Avant d'entamer ce rapport, j'aimerais remercier les gens qui ont été impliqué de loin ou de près dans ce projet.

Tout d'abord, je tiens à remercier mon chef de projet, Monsieur Chenaux, d'avoir accepté d'être mon chef de projet pour ce TPI. Dans la même mesure, je souhaiterais le remercier pour sa disponibilité.

Je souhaite également et particulièrement transmettre mes remerciements aux deux experts, Monsieur Berney et Monsieur Wenger, d'avoir accordé de leur temps, d'avoir contribué à l'amélioration de mon projet grâce à leurs retours.

Finalement, je remercie aussi mes camarades de la FIN2 avec qui j'ai pu échanger concernant ce TPI et qui ont participé à la bonification de mon travail.

## Introduction

Dans le cadre de ma formation d'informaticien et dans le but d'obtenir mon Certificat Fédéral de Capacité d'informaticien, un Travail Personnel Individuel est à réaliser pour conclure mon apprentissage. L'objectif étant de mettre en pratique ce que j'ai appris durant ces deux années passées à l'Ecole des Métiers – Ecole Technique de Lausanne.

Ayant la possibilité de choisir dans quel domaine nous souhaitons faire ce projet, j'ai décidé de me lancer dans un projet de « Développement Desktop » car c'est dans cela que je souhaite me spécialiser par la suite. En effet, aspirant à poursuivre mes études en Informatique Logicielle à l'HEIG-VD, j'ai vu en ce projet une opportunité de me replonger dans la programmation.

### Présentation de l'application

L'application que je suis amené à développer est une To Do List avancée. Le programme doit donner la possibilité à l'utilisateur de se créer un compte et de s'y connecter. L'utilisateur doit pouvoir ajouter des tâches avec une certaine quantité d'informations et les gérer à la façon de la méthode Kanban avec des colonnes. La gestion des tâches à travers les colonnes doit se faire via le Glisser-Déposer. Le logiciel doit également offrir un aspect personnalisable : l'utilisateur doit pouvoir ajouter et supprimer des colonnes, renommer le titre de ces dernières et il doit être en mesure de redimensionner le logiciel comme il le souhaite sans que cela entrave l'interface et les fonctionnalités de l'application.



Figure 1 : Méthode Kanban

# 1 Planification initiale

Cette planification initiale a été la première étape de ce projet. Elle a été réalisée lors du premier jour de TPI.

Avant tout, je souhaiterais préciser que ma planification initiale inclut des plages de 1h30 les vendredis, intitulées « Révision globale du code ».

Ces sessions sont initialement prévues pour être dédiées à l'optimisation du code, revoir des parties éventuellement obsolètes, ... cependant, selon les besoins, ces 1h30 peuvent être également utilisées pour rattraper tout retard dans d'autres aspects du projet, comme la rédaction du rapport.

Séquence 1			Date: lundi, 13 mai 2024		Matin
Tâche	Tranche [5min]		Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...	
Réunion	3	0h15min	Découverte et validation du Cahier des Charges		
Réunion	5	0h25min	Discussion avec Monsieur Wenger (expert)		
Autres	8	0h40min	"Mise en place" du projet : création du GitHub, mise en page des fichiers (JdF, rapport, planif, ...)		
Analyse - Planification initiale	30	2h30min	Rédaction de la planification initiale		
Doc - Journal de travail	2	0h10min	Rédaction du journal de travail		
Total tranche	48	4h			

Séquence 2			Date: lundi, 13 mai 2024		Après-midi
Tâche	Tranche [5min]		Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...	
Analyse - Planification initiale	27	2h15min	Rédaction de la planification initiale		
Doc - Journal de travail	1	0h5min	Rédaction du journal de travail		
Total tranche	28	2h20min			

Séquence 3			Date: mardi, 14 mai 2024		Matin
Tâche	Tranche [5min]		Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...	
Analyse - Documentation	24	2h	Choix de la méthode de gestion de projet, mise en place d'une stratégie de test, analyse du CDC (architecture MVC ?)		
Doc - Rapport	23	1h55min	Création et rédaction du rapport		
Doc - Journal de travail	1	0h5min	Rédaction du journal de travail		
Total tranche	48	4h			

Séquence 4			Date: mardi, 14 mai 2024		Après-midi
Tâche	Tranche [5min]		Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...	
Absences - imprévus	12	1h	Rendez-vous médical (IRM)		
Implémentation - Base de données	15	1h15min	Modélisation (MCD, MLD) de la base de données (sur Looping)		
Doc - Journal de travail	1	0h5min	Création de la base de données (MySQL)		
			Rédaction du journal de travail		
Total tranche	28	2h20min			

Séquence 5			Date: mercredi, 15 mai 2024		Matin
Tâche	Tranche [5min]		Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...	
Doc - Rapport	15	1h15min	Rédaction du rapport		
Implémentation - Interfaces	12	1h	Création de l'interface du formulaire de création d'un compte		
Doc - Journal de travail	1	0h5min	Rédaction du journal de travail		
Total tranche	28	2h20min			

Séquence 6			Date: jeudi, 16 mai 2024		Matin
Tâche	Tranche [5min]		Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...	
Implémentation - Fonctionnalités	12	1h	Implémentation d'une méthode de hashage		
Implémentation - Fonctionnalités	12	1h	Implémentation de la fonctionnalité de création d'un compte		
Tests	8	0h40min	Tests de la fonctionnalité (hashage, sécurité, insertion, ...)		
Implémentation - Interfaces	12	1h	Création de l'interface du formulaire de login (connexion à un compte)		
Implémentation - Fonctionnalités	3	0h15min	Implémentation du login (hashage, sécurité, ...)		
Doc - Journal de travail	1	0h5min	Rédaction du journal de travail		
Total tranche	48	4h			



Séquence 7			Date: jeudi, 16 mai 2024		Après-midi
Tâche	Tranche [5min]		Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...	
Absences - imprévus	12	1h	Rendez-vous médical (CHUV)		
Implémentation - Interfaces	9	0h45min	Implémentation du login (hashage, sécurité, ...)		
Tests	8	0h40min	Test de la fonctionnalité de login (sécurité, insertion, ...)		
Implémentation - Interfaces	8	0h40min	Création de l'interface principale de l'application + responsive (redimensionnement)		
Doc - Journal de travail	1	0h5min	Rédaction du journal de travail		
Total tranche	38	3h10min			

Séquence 8			Date: vendredi, 17 mai 2024		Matin
Tâche	Tranche [5min]		Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...	
Implémentation - Interfaces	24	2h	Suite de la création de l'interface principale de l'application + responsive (redimensionnement)		
Doc - Rapport	23	1h55min	Rédaction du rapport		
Doc - Journal de travail	1	0h5min	Rédaction du journal de travail		
Total tranche	48	4h			

Séquence 9			Date: vendredi, 17 mai 2024		Après-midi
Tâche	Tranche [5min]		Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...	
Implémentation - Révisions	18	1h30min	Révision globale du code (ajustements nécessaires, code incohérent, imprévues ...)		
Implémentation - Fonctionnalités	12	1h	Implémentation de la fonctionnalité permettant de renommer les colonnes (formulaire ? Possibilités ?)		
Tests	7	0h35min	Tests de la fonctionnalité		
Doc - Journal de travail	1	0h5min	Rédaction du journal de travail		
Total tranche	38	3h10min			

Séquence 10			Date: mardi, 21 mai 2024		Matin
Tâche	Tranche [5min]		Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...	
Implémentation - Fonctionnalités	39	3h15min	Implémentation de la fonctionnalité permettant d'ajouter de nouvelles colonnes (placements l, responsive)		
Tests	8	0h40min	Tests de la fonctionnalité		
Doc - Journal de travail	1	0h5min	Rédaction du journal de travail		
Total tranche	48	4h			

Séquence 11			Date: mardi, 21 mai 2024		Après-midi
Tâche	Tranche [5min]		Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...	
Implémentation - Fonctionnalités	27	2h15min	Implémentation de la fonctionnalité permettant de supprimer une colonne (placements des colonnes ?)		
Doc - Journal de travail	1	0h5min	Rédaction du journal de travail		
Total tranche	28	2h20min			

Séquence 12			Date: mercredi, 22 mai 2024		Matin
Tâche	Tranche [5min]		Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...	
Tests	8	0h40min	Test de la fonctionnalité permettant de supprimer une colonne		
Doc - Rapport	19	1h35min	Rédaction du rapport		
Doc - Journal de travail	1	0h5min	Rédaction du journal de travail		
Total tranche	28	2h20min			

Séquence 13			Date: jeudi, 23 mai 2024		Matin
Tâche	Tranche [5min]		Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...	
Implémentation - Fonctionnalités	18	1h30min	Création de l'objet Tâche		
Implémentation - Interfaces	18	1h30min	Création du formulaire d'ajout d'une nouvelle tâche (? image, hyperlien)		
Implémentation - Fonctionnalités	11	0h55min	Implémentation de l'ajout d'une nouvelle tâche		
Doc - Journal de travail	1	0h5min	Rédaction du journal de travail		
Total tranche	48	4h			

Séquence 14			Date: jeudi, 23 mai 2024		Après-midi
Tâche	Tranche [5min]		Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...	
Implémentation - Fonctionnalités	3	0h15min	Suite de l'implémentation permettant l'ajout d'une nouvelle tâche		
Tests	8	0h40min	Test de la fonctionnalité		
Implémentation - Fonctionnalités	26	2h10min	Implémentation du Drag & Drop d'une image dans une tâche (format ?, resize de l'image)		
Doc - Journal de travail	1	0h5min	Rédaction du journal de travail		
Total tranche	38	3h10min			

Séquence 15			Date: vendredi, 24 mai 2024		Matin
Tâche	Tranche [5min]		Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...	
Implémentation - Fonctionnalités	4	0h20min	Suite de l'implémentation du Drag & Drop d'une image		
Tests	8	0h40min	Test de la fonctionnalité		
Doc - Rapport	35	2h55min	Rédaction du rapport		
Doc - Journal de travail	1	0h5min	Rédaction du journal de travail		
Total tranche	48	4h			

Séquence 16			Date: vendredi, 24 mai 2024		Après-midi
Tâche	Tranche [5min]		Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...	
Implémentation - Révisions	18	1h30min	Révision globale du code (ajustements nécessaires, code incohérent, ...)		
Doc - Rapport	19	1h35min	Rédaction du rapport		
Doc - Journal de travail	1	0h5min	Rédaction du journal de travail		
Total tranche	38	3h10min			

Séquence 17			Date: lundi, 27 mai 2024		Matin
Tâche	Tranche [5min]		Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...	
Implémentation - Fonctionnalités	47	3h55min	Implémentation du Drag & Drop entre colonnes		
Doc - Journal de travail	1	0h5min	Rédaction du journal de travail		
Total tranche	48	4h			

Séquence 18			Date: lundi, 27 mai 2024		Après-midi
Tâche	Tranche [5min]		Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...	
Tests	12	1h	Test de la fonctionnalité		
Implémentation - Révisions	15	1h15min	Révision du code dans son intégralité - optimisation		
Doc - Journal de travail	1	0h5min	Rédaction du journal de travail		
Total tranche	28	2h20min			

Séquence 19			Date: mardi, 28 mai 2024		Matin
Tâche	Tranche [5min]		Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...	
Implémentation - Révisions	21	1h45min	Suite de la révision du code - optimisation		
Tests	26	2h10min	Tests de toutes les fonctionnalités		
Doc - Journal de travail	1	0h5min	Rédaction du journal de travail		
Total tranche	48	4h			

Séquence 20			Date: mardi, 28 mai 2024		Après-midi
Tâche	Tranche [5min]		Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...	
Tests	27	2h15min	Suite des tests		
Doc - Journal de travail	1	0h5min	Rédaction du journal de travail		
Total tranche	28	2h20min			

Séquence 21			Date: mercredi, 29 mai 2024		Matin
Tâche	Tranche [5min]		Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...	
Doc - Rapport	27	2h15min	Rédaction du rapport		
Doc - Journal de travail	1	0h5min	Rédaction du journal de travail		
Total tranche	28	2h20min			

Séquence 22			Date: jeudi, 30 mai 2024		Matin
Tâche	Tranche [5min]		Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...	
Doc - Rapport	47	3h55min	Rédaction du rapport		
Doc - Journal de travail	1	0h5min	Rédaction du journal de travail		
Total tranche	48	4h			

Séquence 23			Date: jeudi, 30 mai 2024		Après-midi
Tâche	Tranche [5min]		Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...	
Doc - Rapport	37	3h5min	Rédaction du rapport		
Doc - Journal de travail	1	0h5min	Rédaction du journal de travail		
Total tranche	38	3h10min			

Séquence 24			Date: vendredi, 31 mai 2024		Matin
Tâche	Tranche [5min]		Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...	
Doc - Rapport	47	3h55min	Rédaction du rapport		
Doc - Journal de travail	1	0h5min	Rédaction du journal de travail		
Total tranche	48	4h			

Séquence 25			Date: vendredi, 31 mai 2024		Après-midi
Tâche	Tranche [5min]		Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...	
Doc - Rapport	37	3h5min	Rédaction du rapport		
Doc - Journal de travail	1	0h5min	Rédaction du journal de travail		
Total tranche	38	3h10min			

Séquence 26			Date: lundi, 3 juin 2024		Matin
Tâche	Tranche [5min]		Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...	
Doc - Rapport	47	3h55min	Rédaction du rapport		
Doc - Journal de travail	1	0h5min	Rédaction du journal de travail		
Total tranche	48	4h			

Séquence 27			Date: lundi, 3 juin 2024		Après-midi
Tâche	Tranche [5min]		Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...	
Doc - Rapport	16	1h20min	Rédaction du rapport		
Doc - Journal de travail	6	0h30min	Rédaction du journal de travail		
Autres	6	0h30min	FIN DU TPI - Envoi des livrables (! Nommage, formats, adresse mail !)		
Total tranche	28	2h20min			



## 2 Analyse

### 2.1 Planification détaillée

### 2.2 Méthodologie

#### 2.2.1 Méthode de gestion de projet

La méthode de gestion de projet « Waterfall », plus particulièrement la méthode des « 6 pas » a été choisie et va être appliquée à ce projet.

Les 6 étapes de cette méthode sont les suivantes :

#### **S'informer – Planifier – Décider – Réaliser – Contrôler – Evaluer**

Le cahier des charges (par conséquent, les objectifs du projet également) ayant clairement été défini et validé au lancement du projet ainsi que la date de rendue, la méthode des 6 pas s'est rapidement présentée comme une solution fiable, envisageable et adaptée à la situation.

De plus, la structure des canevas fournis pour le journal de travail, la planification ainsi que le rapport correspond à la méthodologie des 6 pas. Par exemple, la tâche « Analyse – Planification initiale » s'inscrit totalement dans les étapes « S'informer, Planifier et Décider ».

En outre, étant une personne que l'on peut qualifier de perfectionniste, cette méthode structurée me permettra de m'éviter de tomber dans un peaufinage excessif des détails, qui conduirait à une mauvaise gestion du temps et entraînerait un potentiel retard. De ce fait, la progression linéaire et segmentée des 6 pas me servira de fil conducteur et m'encadrera dans mon travail.

#### **Concrètement, comment cette méthode va s'appliquer à mon projet ?**

J'ai d'abord élaboré une planification initiale qui, implicitement, suit les 6 pas. En effet, comme je l'ai déjà expliqué précédemment, la tâche « Analyse – Planification initiale » est la parfaite représentation de cette méthodologie, car pour faire cette planification, j'ai commencé par lire et analyser le cahier des charges (s'informer), j'ai ensuite dû faire ma planification initiale (planifier) et en même temps choisir (décider) l'ordre dans lequel j'allais m'y prendre, par quelles fonctionnalités j'allais commencer...

J'ai ensuite séparé les fonctionnalités comme des « mini-projets » et à chaque mini-projet la méthode des 6 pas est appliquée.

Par exemple, j'ai fait de la « Possibilité de se connecter de manière sécurisée avec des comptes utilisateurs. », qui est une fonctionnalité demandée dans le cahier des charges, un mini-projet.

**S'informer** : Cette étape consiste à comprendre ce qu'il y a à réaliser et prendre conscience des défis à relever. La documentation sur internet peut déjà commencer à ce stade si j'estime que c'est utile.

#### Qu'est-ce que cette fonctionnalité implique ?

La création d'un formulaire permettant à l'utilisateur de se logger et un formulaire permettant de créer un compte si l'utilisateur n'a pas de compte.

Y a-t-il potentiellement des points qui vont me poser un ou des problèmes ?

L'aspect « sécurité » de la fonctionnalité (exécution des requêtes sécurisée, utiliser quelle méthode de hachage du mot de passe, ...) et l'aspect « Notification » du modèle MVC (valider qu'une donnée a été insérée correctement dans la base de données et avertir l'utilisateur via la vue).

Ces points problématiques sont-ils importants ?

L'aspect sécurité l'est. Tandis que l'aspect « Notification » l'est moins, car il n'est pas mentionné dans le cahier des charges, c'est uniquement une contrainte de l'architecture MVC.

Comment puis-je résoudre ces problèmes ?

En me documentant sur internet, en utilisant ChatGPT, en demandant de l'aide à un camarade ou à mon chef de projet.

**Planifier et Décider :** C'est le moment de mettre en place ma planification détaillée pour ce mini-projet sur laquelle je vais me baser pour la partie « Réaliser ».

Si j'ai détecté de potentiels problèmes à la question précédente, combien de temps leurs résolutions risquent de me prendre ?

Cela risque de prendre une heure supplémentaire.

Quelles sont les priorités ?

D'abord, l'utilisateur doit pouvoir se connecter à un compte existant et doit également pouvoir créer un nouveau compte. Ensuite, la solution doit fonctionner de manière sécurisée. Pour finir, le code doit être optimisé.

Combien de temps maximum j'accorde à la réalisation de cette fonctionnalité (mini-projet) ?

Jusqu'au vendredi 17 mai.

**Décider :** Ce sont les décisions importantes à prendre avant de se lancer dans la réalisation afin de structurer sa façon de faire.

S'il y a plusieurs solutions pour la résolution des problèmes pressentis, quelle(s) méthode(s) vais-je utiliser ?

A propos de la recherche de solutions, je vais surtout me documenter sur internet. Si cela ne suffit pas, j'utiliserai ChatGPT et si cela n'est pas encore suffisant, je demanderai de l'aide à mon chef de projet. Concernant le hachage du mot de passe, je pense utiliser BCrypt et mes recherches internet me confirmeront mon choix.

Quels sont les critères de réussite ?

L'utilisateur doit pouvoir créer un compte et se connecter et le processus doit être sécurisé et doit au minimum protéger contre les injections SQL et haché le mot de passe.

**Réaliser :** La réalisation est le cœur de la tâche. C'est la pratique.

**Contrôler :** Le contrôle est l'étape de test. C'est le moment de s'assurer que ce qui a été fait dans la partie « Réaliser » répond aux attentes et aux objectifs.

La solution a-t-elle été validée par les tests de validation de la stratégie de tests ?

**Evaluer** : Cette partie permet de faire une courte rétrospective ce qui vient d'être fait qui permettra de déterminer plusieurs choses.

Y a-t-il des objectifs supplémentaires que je peux considérer si j'ai davantage de temps à ma disposition ?

La façon dont je traite les « notifications (MVC) », que j'ai laissé de côté, pourrait être revue et optimisée (code répétitif). Le code pourrait globalement être optimisé.

A quel point ces objectifs supplémentaires sont-ils importants ? Vont-ils réellement amener une plus-value au projet ?

L'optimisation du code pourrait rendre le tout plus simple à gérer, mais ce n'est pas primordial car la solution reste fonctionnelle et sans cela.

Si j'ai perdu du temps, où et pourquoi ai-je perdu du temps ?

La conception des maquettes des interfaces m'a fait « perdre » du temps car je n'avais pas prévu du temps pour cela dans ma planification. J'ai également sous-estimé le temps que la recherche de documentation sur internet m'a prise.

## 2.2.2 Sauvegarde

La perte de données est un risque omniprésent, encore plus dans un projet comme celui-ci et plus généralement dans le métier d'informaticien. Il est crucial de hautement considérer ce danger.

C'est pourquoi j'ai décidé d'établir une « routine » de backup facile à mettre en place afin de minimiser ce risque sans que cela ne soit trop coûteux en termes de temps.

**La routine est la suivante :**

- **Utilisation de GitHub** : GitHub, en plus de permettre le versioning et d'optimiser la portabilité du projet, est une véritable solution de sauvegarde viable. C'est pour ces raisons que j'ai décidé de créer mon projet « actif » sur GitHub.
- **Push régulier sur GitHub** : Au minimum, avant chaque pause ou fin de journée (toutes les 1h30), un commit et un push sont effectués, afin de garantir que les dernières modifications ont été sauvegardées.
- **Sauvegarde sur SSD et en local** : En complément de GitHub, une copie du TPI est également sauvegardée sur mon disque dur externe ainsi que sur l'ordinateur de travail, en local, après chaque mise à jour sur GitHub.

## 2.3 Conception

### 2.3.1 Modélisation de la base de données

Avant de me lancer dans la création et l'écriture du script SQL de ma base de données, j'ai appliqué et suivi la méthode MERISE pour modéliser ma base de données afin de m'assurer de sa cohérence et qu'elle pourra supporter mon application.

MERISE, qui signifie Méthode d'Etude et de Réalisation Informatique pour les Systèmes d'Entreprise, est une méthode d'analyse, de conception et de gestion de projet, surtout utilisée dans le monde de l'informatique.

Dans le cadre de l'élaboration d'une base de données, MERISE propose des « outils » permettant de réaliser une base de données de manière méthodique. Suivre cette méthodologie consiste à d'abord élaborer un Modèle Conceptuel des Données (MCD), qui est un schéma qui met particulièrement en évidence comment les éléments sont liés entre eux, comme le démontre la présence de « cardinalités ». Par exemple, dans le schéma ci-dessous, les cardinalités indiquent que chaque utilisateur (USER) peut créer zéro ou plusieurs tâches (TASK), alors qu'une tâche doit être créée par un seul utilisateur. Cela souligne une dépendance : une tâche ne peut pas exister sans un utilisateur. Cette observation, nous le verrons plus tard, sera déterminante pour le développement de l'application. En bref, Le MCD est étape analytique qui permet de faire une première « ébauche » de la structure de la base de données à partir de ce qui est demandé (dans mon cas, par le cahier des charges).

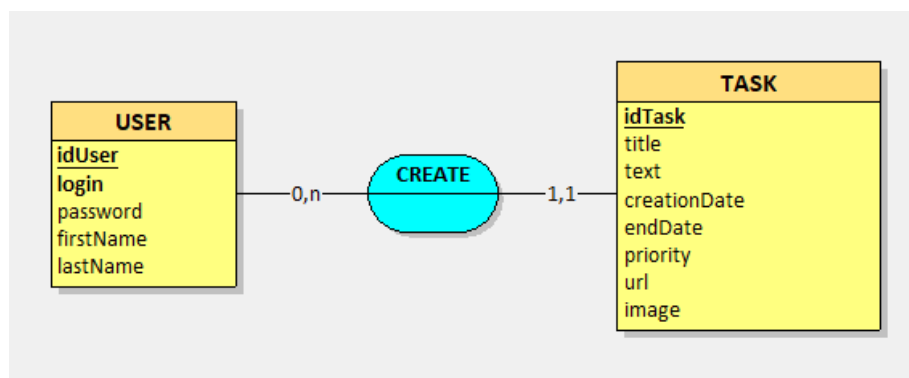


Figure 2 : Modèle Conceptuel des Données

Une fois que le MCD est terminé, le Modèle Logique des Données (MLD) peut être construit. Le MLD reprend le contenu du MCD et le transpose dans un format plus « technique », se rapprochant bien plus du résultat final qui sera prêt à être implémenté dans un Système de Gestion de Base de Données. Cela se reflète notamment dans le nom des attributs comme « useLogin » ou encore dans les entités TASK et USER qui deviennent des tables et les associations (CREATE), qui représentent les relations entre les entités, qui sont traduites en clés étrangères. Tous ces changements seront conservés lors de l'implémentation dans la base de données.

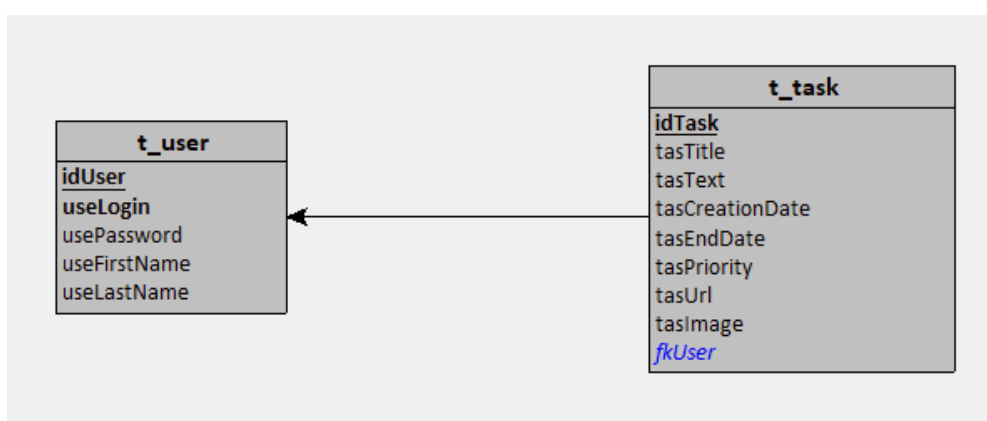


Figure 3 : Modèle Logique des Données (fait sur Looping)

Le Modèle Physique des données (MPD) est la dernière étape de la méthode MERISE dans la conception d'une base de données. Il représente concrètement la structure de la base de données en détaillant notamment les types de données (int, varchar, ...). Contrairement au

MCD et au MLD, le MPD est spécifique au système de gestion de base de données, ici SQL.

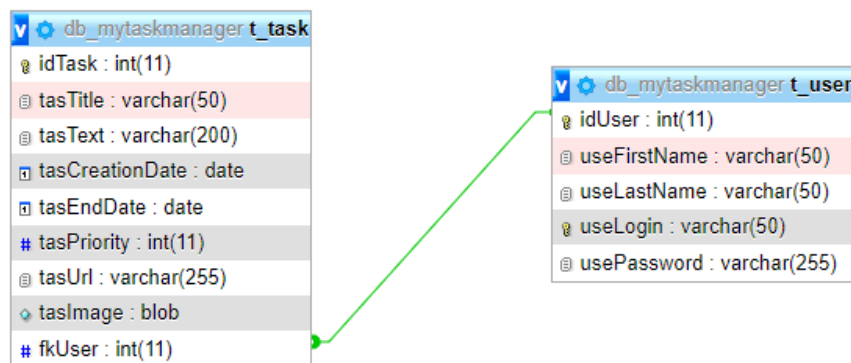


Figure 4 : Modèle Physique des Données (représentation depuis PHPMyAdmin)

### Explication des spécificités de la table User :

useLogin est unique. Cela signifie que deux comptes utilisateurs ne peuvent pas posséder le même login. J'ai appliqué cette propriété pour éviter que deux comptes partageant exactement les mêmes informations puissent coexister dans la base de données, afin d'empêcher les brèches de sécurité que cela pourrait provoquer.

### Explication des spécificités de la table Task :

tasCreation et tasEndDate ont pour type d'attributs « Date » et non pas « DateTime » car je souhaite uniquement stocker des dates, et non pas des dates avec les heures.

tasUrl est un « blob » car je souhaite stocker les images directement dans la base de données malgré le fait que cela puisse potentiellement ralentir les performances de cette dernière. Tout d'abord, parce que cette façon de faire est la plus simple à gérer. De plus, dans son état actuel, MyTaskManager est un « petit projet ». Stocker directement les images ne risque pas d'avoir de grande conséquence sur les performances et sur l'expérience de l'utilisateur. De plus, comme je vous l'expliquerai dans la pratique, j'ai tenté de limiter l'espace de stockage que prennent les images en portant une attention sur le format de ces dernières.

## 2.3.2 UwAmp

UwAmp est un serveur Wamp léger et portable permettant d'exécuter Apache, PHP, PHPMyAdmin, etc. qui donne la possibilité de tester son site web ou encore sa base de données.

J'ai choisi d'utiliser UwAmp en lieu et place de Docker car UwAmp est une solution portable, ne nécessitant aucune installation. A travers les divers projets (PHP, MySQL) que j'ai menés, je me suis familiarisé avec UwAmp et toujours dans cette quête de gestion du temps, j'ai préféré privilégier un logiciel que je connais très bien et qui répond à ce dont j'ai besoin pour réussir ce projet à la place de Docker, qui, malgré ses nombreuses qualités, n'est pas un outil avec lequel j'ai eu l'opportunité de vraiment travailler.

## 2.3.3 Maquettes

Pour établir une cohérence graphique, j'ai réalisé des maquettes de chaque interface de mon application sur Figma.

Le logo, étant un symbole visuel qui représente l'identité d'une marque, d'un logiciel, etc., j'ai, de ce fait, commencé par la conception du logo de mon application. J'ai essayé d'utiliser ChatGPT 4 pour générer un logo correspondant à une application « To Do List », en vain. Les résultats proposés étaient soit trop brouillon ou soit ils ne me satisfaisaient pas malgré les multiples tentatives mais cela m'a permis de mieux visualiser l'identité visuelle que je souhaitais donner à ce projet.



Figure 5 : Post-It

J'ai donc fini par réaliser le logo moi-même sur Canva. Je voulais une image simpliste et parlante. J'ai choisi le jaune comme couleur principale car, au-delà d'être une couleur qui attire l'œil, elle me rappelle les Post-It que beaucoup utilisaient pour noter leurs tâches.

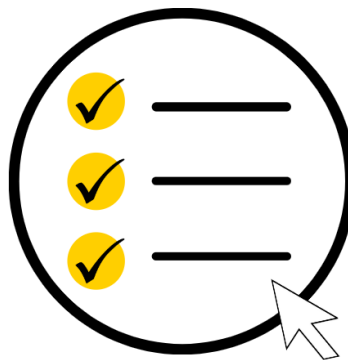


Figure 6 : Logo de MyTaskManager

Les fondamentaux posés, j'ai pu ensuite me lancer dans l'élaboration des différentes interfaces qui composeraient mon application.

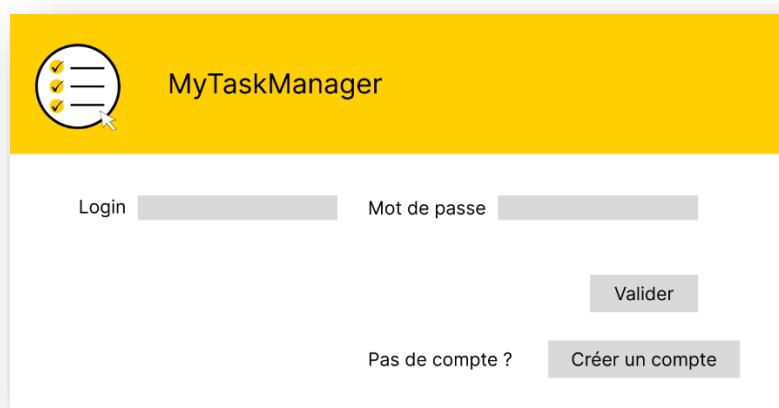
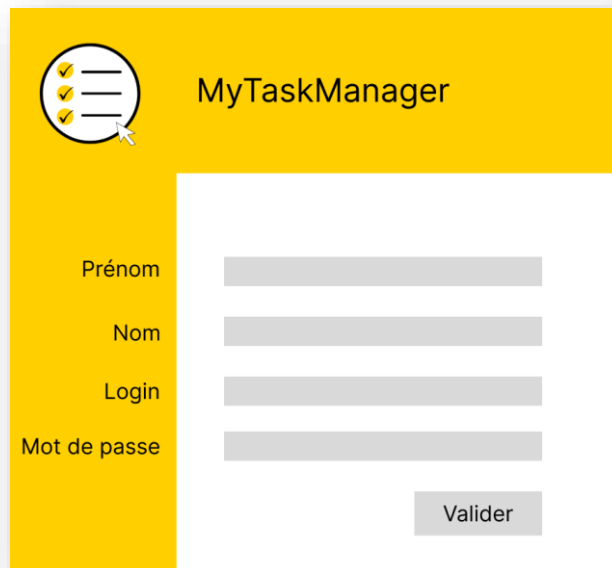


Figure 7 : Maquette de l'interface de connexion



MyTaskManager

Prénom

Nom

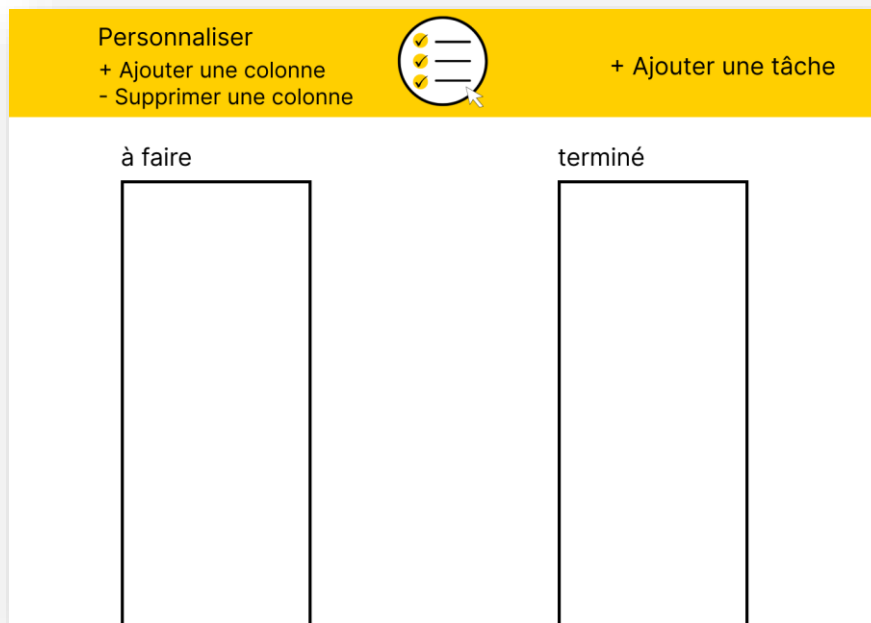
Login

Mot de passe

Valider

Figure 8 : Maquette de l'interface de création d'un compte

Pour ce qui est de l'interface principale, j'ai rapidement réalisé deux concepts. La version 1 ne me convenait pas visuellement et allait probablement être difficile à gérer, notamment par rapport à l'aspect *personnalisation* de l'application avec la gestion des colonnes. C'est pour ces raisons que mon choix s'est finalement porté sur le deuxième concept.



Personnaliser  
+ Ajouter une colonne  
- Supprimer une colonne

+ Ajouter une tâche

à faire

terminé

Figure 9 : Maquette de l'interface principale (version 1)

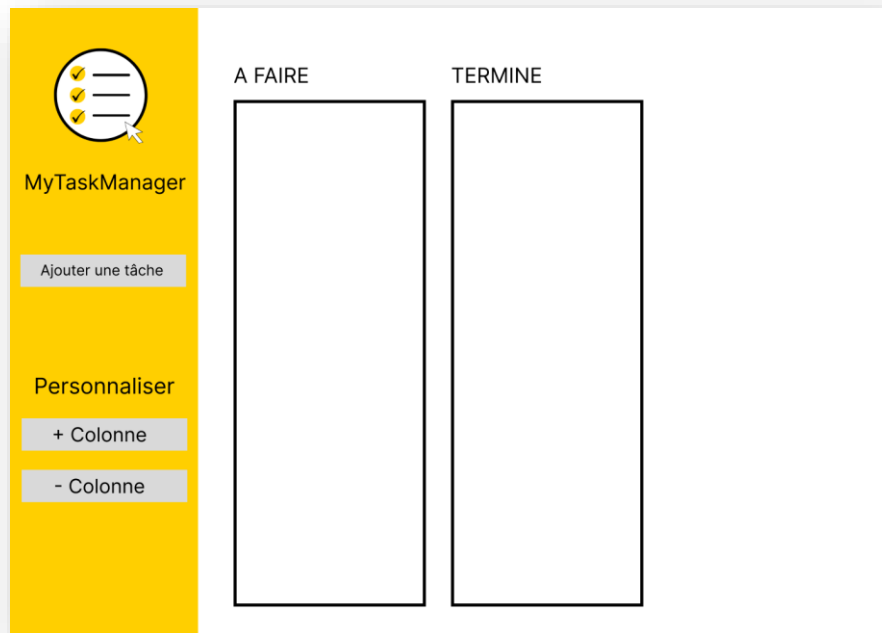


Figure 10 : Maquette de l'interface principale (version finale)



Figure 11 : Maquette de l'interface d'ajout d'une tâche



### 2.3.4 Architecture MVC

MVC est une architecture logicielle qui permet de séparer le développement d'une application en 3 « composants » distincts : **Modèle**, **Vue** et **Contrôleur**.

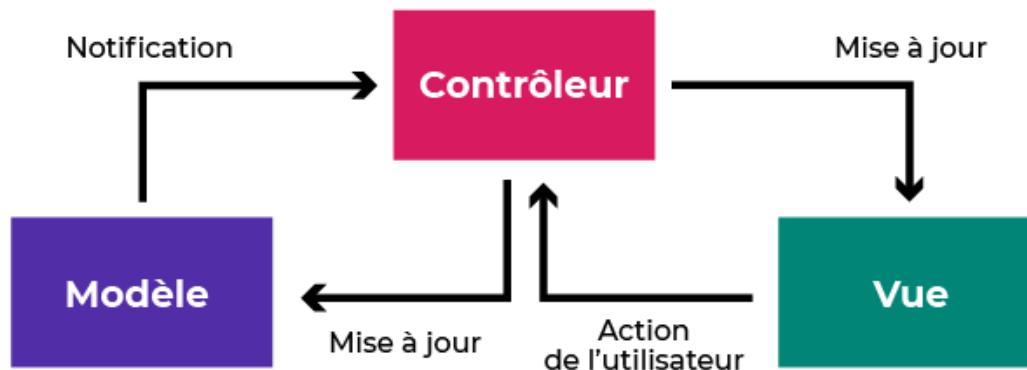


Figure 12 : Fonctionnement du MVC schématisé

**Chaque composant se charge d'un aspect spécifique de l'application.**

**Le Modèle :** Il contient les données manipulées par le programme. Il assure la gestion de ces données et garantit leur intégrité. Dans le cas typique d'une base de données, c'est le modèle qui a accès à la base de données<sup>1</sup>. Concrètement, il contient les opérations servant à manipuler les données, en incluant, par exemple, une méthode permettant d'ajouter (créer) un nouvel utilisateur.

**La Vue :** Elle gère toute la logique interface utilisateur. Elle génère l'interface que l'utilisateur va voir et utiliser. Elle est chargée d'afficher les données (fournies par le Modèle) et de recevoir les actions de l'utilisateur (clic de souris, remplissage d'un champ, etc.).

**Le Contrôleur :** C'est l'intermédiaire entre la Vue et le Modèle. Le contrôleur reçoit les événements enclenchés par l'utilisateur depuis la Vue et avertit, par conséquent, le modèle en lui indiquant quoi faire. Inversement, le modèle notifie le contrôleur des changements pour que ce dernier mette à jour la Vue.

Le pattern MVC étant la seule architecture logicielle que l'on a réellement étudiée et dans l'intérêt de « structurer » le code, ce modèle MVC a donc logiquement été utilisé et suivi pour le développement de l'application.

Voici comment j'ai organisé mon MVC :

<sup>1</sup> Explication de l'architecture MVC :

<https://www.irif.fr/~carton/Enseignement/InterfacesGraphiques/Cours/Swing/mvc.html>

Toujours dans l'optique de structurer le code et de répartir spécifiquement les aspects de l'application, j'ai opté pour une séparation entre la partie Utilisateur et la partie Tâche du programme. C'est pourquoi j'ai deux Contrôleurs (MyTaskController et UserController) ainsi que deux Modèles (MyTaskModel et UserModel).

De plus, il faut noter la présence de l'objet Colonne dans les Vues. Cet objet sert uniquement à l'interface principale de l'application, permettant à l'utilisateur de gérer les tâches. Les tâches s'affichant dans les colonnes, j'ai spécialement donné l'accès à l'objet Colonne à l'objet Tâche (MyTask) malgré le fait que cela ne respecte pas le principe de l'architecture MVC. Colonne étant un objet uniquement visuel, cela n'aura pas d'impact sur l'implémentation du pattern MVC dans le reste du programme.

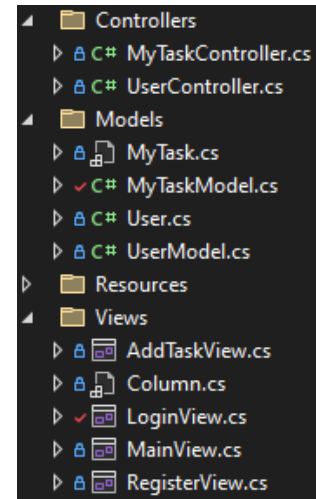


Figure 13 : MVC de MyTaskManager

## 2.4 Stratégie de test

Pour garantir le succès d'un projet, il est fondamental de s'assurer que le résultat du travail accompli soit fonctionnel. Pour ce faire, il faut réaliser des tests. Ces tests permettent de tester les limites de la solution et de détecter d'éventuelles erreurs. Tester est laborieux car il y a de nombreux types de test qui existent et chaque projet nécessite une façon de tester spécifique. C'est pourquoi il est important de planifier une stratégie de tests efficace et adaptée, permettant de couvrir l'étendue de l'ampleur du projet.

Dans le cadre du développement d'une application fonctionnant avec une base de données SQL, il est important de contrôler trois éléments majeurs : fonctionnement des fonctionnalités, sécurité, optimisation de l'expérience utilisateur.

Les fonctionnalités demandées par le cahier des charges doivent être fonctionnelles et répondre aux critères de satisfaction.

La sécurité doit être primordiale car des données privées vont être potentiellement stockées dans la base de données.

L'expérience utilisateur ne doit pas être entravée par des bugs ou par manque de facilité d'utilisation.

Pour définir clairement l'objectif de mes tests, j'ai décidé d'utiliser le principe des « User Stories ». Un récit utilisateur ou User Story est une phrase servant à expliquer une fonctionnalité logicielle, rédigée du point de vue de l'utilisateur final<sup>2</sup>.

<sup>2</sup> Définition des User Stories : <https://asana.com/fr/resources/user-stories>

## 3 Réalisation

Le chapitre « Réalisation » va couvrir toute la partie « Pratique » de ce projet. A travers ce chapitre, je vais expliquer le fonctionnement du code de mon application, raconter et justifier mes choix pour vous donner un aperçu de ma façon de faire et de penser.

J'aimerais d'abord clarifier la chronologie de la réalisation. L'application MyTaskManager peut être séparée en deux parties : gestion des utilisateurs et gestion des tâches.

Nous avons constaté durant la partie *Analyse*, lors du chapitre *Conception de la base de données*, qu'il y avait une dépendance : une tâche ne peut pas exister sans un utilisateur. De ce fait, je me suis d'abord occupé de programmer la partie *Utilisateur* de l'application. La gestion des tâches se fera uniquement lorsque les fonctionnalités liées à la gestion des utilisateurs seront fonctionnelles.

De manière plus générale et à chaque implémentation d'une nouvelle fonctionnalité, j'ai constamment procédé de la même façon : je commence par la Vue (faire l'interface s'il y en a une), je continue avec le Contrôleur et je finis avec le Modèle. C'est pourquoi je vais structurer de la même manière les paragraphes explicatifs de chaque fonctionnalité.

### 3.1 Compte utilisateur

#### 3.1.1 Hachage

Les comptes utilisateurs contenant des informations particulièrement sensibles, notamment comme un mot de passe, il est indispensable de protéger et sécuriser ces données.

C'est là que le hachage intervient. En effet, le hachage est un processus unidirectionnel qui permet de convertir une donnée comme un mot de passe en texte chiffré à l'aide « d'algorithmes de hachage »<sup>3</sup>.

J'ai opté pour l'utilisation de la bibliothèque BCrypt, principalement car elle offre une méthode de hachage spécialement conçue et adaptée pour les mots de passe. BCrypt est pratique en termes de développement car elle offre une méthode de hachage. Il suffit d'appeler cette méthode préconçue et le hachage est fait, sans que j'aie eu besoin de me charger de la création de ma propre méthode. Au-delà de l'aspect « pratique », BCrypt utilise également le salage dans sa fonction de hachage. Le salage ajoute des caractères aléatoires au hachage, assurant ainsi que chaque hash généré est unique. Par exemple, dans le cas où deux utilisateurs ont le même mot de passe, BCrypt produit deux hashes distincts, ce qui renforce la sécurité et permet d'éviter les « attaques par tables arc-en-ciel »<sup>4</sup>.

```
string hashedPassword = BCrypt.Net.BCrypt.HashPassword(newUser.Password);
```

Figure 14 : Méthode de hachage BCrypt

<sup>3</sup> Définition du hachage de Mail in Black :

<https://www.mailinblack.com/ressources/blog/le-salage-de-mots-de-passe-une-couche-de-securite-indispensable/>

<sup>4</sup> Pourquoi BCrypt est une solution pour le hachage des mots de passe :

<https://nordvpn.com/fr/blog/what-is-bcrypt/>

### 3.1.2 Création de compte

Comme je l'ai déjà mentionné précédemment, mon application propose la possibilité de créer un compte utilisateur.

Lorsque l'utilisateur a rempli le formulaire ([design du formulaire](#)) de création d'un compte, au clic du bouton « Valider », l'action est récupérée par la vue et les informations des champs du formulaire sont envoyées au Contrôleur des utilisateurs.

```
private void btnCreateUser_Click(object sender, EventArgs e)
{
    // Transmets les informations insérées au Contrôleur et vérifie si la création du nouveau compte utilisateur a fonctionné
    if(UserController.RegisterUser(tbxFirstName.Text, tbxLastName.Text, tbxLogin.Text, tbxPassword.Text).Item1)
    {
        MessageBox.Show("Nouveau compte utilisateur crée avec succès !", "Création d'un compte");
        this.Close();
    }
    else
    {
        string errorMessage = UserController.RegisterUser(tbxFirstName.Text, tbxLastName.Text, tbxLogin.Text, tbxPassword.Text).Item2.Message;
        MessageBox.Show("Erreur : " + errorMessage, "Création de compte");
    }
}
```

Figure 15 : Événement au clic du bouton « Créer un compte »

Le Contrôleur crée alors un objet Utilisateur à partir des informations récoltées pour l'envoyer au Modèle des utilisateurs.

```
public (bool, Exception) RegisterUser(string firstName, string lastName, string login, string password)
{
    try
    {
        User newUser = new User(firstName, lastName, login, password);

        // Transmets les informations du nouveau compte utilisateur au Modèle
        return _model.RegisterUserInDatabase(newUser);
    }
    catch (Exception ex)
    {
        return (false, ex);
    }
}
```

Figure 16 : Méthode « Créer un compte » dans le Contrôleur

Le Modèle se charge d'ouvrir la connexion entre l'application et la base de données et, après avoir hashé le mot de passe comme mentionné dans le chapitre précédent, de préparer la requête SQL que la base de données va exécuter pour ajouter le nouvel utilisateur.

Afin de protéger mon application et ma base de données des attaques par injections SQL, les valeurs ne sont pas directement injectées dans la requête SQL. A la place, j'ai utilisé des paramètres SQL (@useFirstName). J'appelle ensuite la méthode « AddWithValue » qui permet de lier le paramètre SQL avec la valeur correspondante. Cette étape renforce la sécurité de l'application.

```
public (bool,Exception) RegisterUserInDatabase(User newUser)
{
    try
    {
        // Ouvre la connexion à la base de données
        _connection.Open();

        // Stocke et hache le mot de passe
        string hashedPassword = BCrypt.Net.BCrypt.HashPassword(newUser.Password);

        // Prépare la requête SQL pour insérer un nouvel utilisateur dans la base de données
        MySqlCommand query = new MySqlCommand(
            "INSERT INTO t_user VALUES(NULL, @useFirstName, @useLastName, @useLogin, @usePassword)", _connection);

        // Data Binding (permet de faire de la liaison de données et par conséquent d'éviter les injection SQL)
        query.Parameters.AddWithValue("@useFirstName", newUser.FirstName);
        query.Parameters.AddWithValue("@useLastName", newUser.LastName);
        query.Parameters.AddWithValue("@useLogin", newUser.Login);
        query.Parameters.AddWithValue("@usePassword", hashedPassword);

        // Exécute la requête SQL
        int result = query.ExecuteNonQuery();

        // Vérifie si au moins une colonne a été affectée et par conséquent si l'opération a fonctionné
        if (result > 0)
        {
            return (true, null);
        }
        else
        {
            return (false, null);
        }
    }
    catch (Exception ex)
    {
        return (false, ex);
    }
    finally
    {
        // Ferme la connexion indépendamment du résultat de l'opération (succès ou échec)
        _connection.Close();
    }
}
```

Figure 17 : Méthode « Créer un compte » dans le Modèle

Pour notifier l'utilisateur du résultat du processus, chaque méthode dans le Contrôleur ou dans le Modèle retourne un tuple contenant un booléen et une exception. Le booléen définit si l'opération s'est correctement déroulée. Dans le cas contraire, l'erreur est retournée afin de pouvoir l'afficher dans la Vue.

### 3.1.3 Login (Connexion à un compte existant)

Parce que la fonctionnalité de création de compte ne va pas sans la fonctionnalité de connexion à un compte, j'ai implémenté un login.

Après avoir reproduit [la maquette du design du login](#) via le designer et la boîte à outils de Visual Studio, j'ai défini l'événement Clic lorsque l'utilisateur clique sur le bouton *Connexion*. La Vue envoie les informations entrées par l'utilisateur dans le formulaire au Contrôleur des utilisateurs.

Le Contrôleur fait simplement acte d'intermédiaire et transmet les informations au Modèle des utilisateurs.

La méthode LoginUserToDB de mon Modèle agit de la même manière de lors de la création de compte : il ouvre la connexion à la base de données, prépare la requête SQL en la protégeant des injections SQL via la liaison des données (Data Binding) et exécute la requête SQL. A partir de là, il y a des changements notables. La requête SQL est non-seulement exécutée mais également lu par un objet « *MySqlDataReader* » qui permet, comme son nom le suggère, de lire les données récupérées par l'exécution de la requête SQL. Si une ligne est lue avec le login entré par l'utilisateur, cela signifie que le compte utilisateur, auquel

l'utilisateur actuel essaie de se connecter, existe dans la base de données. Alors, le mot de passe haché est récupéré et comparé au mot de passe inséré par l'utilisateur via une méthode de la bibliothèque BCrypt. Si les mots de passe correspondent, l'ID du compte utilisateur est retourné.

```
// Vérifie la correspondance entre les mots de passe
if(BCrypt.Net.BCrypt.Verify(password, hashedPassword))
{
    // Stocke et retourne l'id de l'utilisateur actuel
    int currentUserId = (int)reader["idUser"];
    return (currentUserId, null);
}
else
{
    return (null, null);
}
```

Figure 18 : Méthode BCrypt.Verify()

De nouveau, afin de notifier l'utilisateur du résultat du processus, la méthode du Modèle et du Contrôleur retourne un tuple contenant un entier *nullable* et une exception. L'entier représente l'ID de l'utilisateur s'il a réussi à se connecter, sinon il est *null*. En cas d'erreur, l'exception est retournée afin de pouvoir l'afficher dans la Vue.

```
else
{
    // Vérifie le type d'erreur : Erreur de login/mot de passe ou autre erreur
    if (UserController.LoginUser(tbxLogin.Text, tbxPassword.Text).Item2 != null)
    {
        string errorMessage = UserController.LoginUser(tbxLogin.Text, tbxPassword.Text).Item2.Message;
        MessageBox.Show("Erreur : " + errorMessage, "Connexion de compte");
    }
    else
    {
        MessageBox.Show("Erreur : Login et/ou mot de passe incorrect");
    }
}
```

Figure 19 : Gestion des erreurs lors du login (dans la Vue)

Dans le cas où l'erreur est un mot de passe incorrect, l'utilisateur est informé. En effet, si aucune exception n'est retournée (*null* car il n'y a pas eu d'erreur au niveau du Modèle) et que l'ID de l'utilisateur est également *null*, cela signifie que le problème provient de l'utilisateur qui n'a pas entré le bon mot de passe. Cela permet de faire la distinction entre les erreurs liées aux identifiants de connexion incorrects et les autres erreurs potentielles.

## 3.2 Gestion des tâches

### 3.2.1 Objet Tâche

### 3.2.2 Ajout d'une tâche

### 3.2.3 Affichage des tâches

## 4 Tests

## 5 Conclusion



## 6 Annexes

### 6.1 Table des illustrations

Figure 1 : Méthode Kanban .....	5
Figure 2 : Modèle Conceptuel des Données.....	20
Figure 3 : Modèle Logique des Données (fait sur Looping).....	20
Figure 4 : Modèle Physique des Données (représentation depuis PHPMyAdmin) .....	21
Figure 5 : Post-It.....	22
Figure 6 : Logo de MyTaskManager .....	22
Figure 7 : Maquette de l'interface de connexion .....	22
Figure 8 : Maquette de l'interface de création d'un compte .....	23
Figure 9 : Maquette de l'interface principale (version 1).....	23
Figure 10 : Maquette de l'interface principale (version finale) .....	24
Figure 11 : Maquette de l'interface d'ajout d'une tâche .....	24
Figure 12 : Fonctionnement du MVC schématisé .....	25
Figure 13 : MVC de MyTaskManager .....	26
Figure 14 : Méthode de hachage BCrypt .....	27
Figure 15 : Evénement au clic du bouton « Créer un compte » .....	28
Figure 16 : Méthode « Créer un compte » dans le Contrôleur .....	28
Figure 17 : Méthode « Créer un compte » dans le Modèle .....	29
Figure 18 : Méthode BCrypt.Verify() .....	30
Figure 19 : Gestion des erreurs lors du login (dans la Vue) .....	30